# A Self-diagnosis Algorithm Based on Causal Graphs

Jingxian Lu
Orange Labs
2, Avenue Pierre Marzin
22300 Lannion Cedex, France
jingxian.lu@orange-ftgroup.com

Christophe Dousson
Orange Labs
2, Avenue Pierre Marzin
22300 Lannion Cedex, France
christophe.dousson@orange-ftgroup.com

Francine Krief
LaBRI - University Bordeaux 1
351, Cours de La Liberation
33405 Talence Cedex, France
francine.krief@labri.fr

***Abstract*—The concept of autonomic networking has been introduced to facilitate network management that is increasingly complex. It uses a closed control loop proposed by the autonomic computing. Obviously, this approach relies on a distributed architecture. This implies that the diagnosis algorithms have to be distributed in order to satisfy autonomic architecture requirements. In this paper we describe novel algorithms regarding the use of causal graph model to perform diagnosis distribution in telecommunication networks.**

***Keywords*—diagnosis algorithm; causal graph; distribution; self-diagnosis.**

## I. Introduction

With the growth of network complexity and heterogeneity, the need for managing and controlling the network effectively by reducing human intervention seems essential. An autonomic system can monitor itself and adjust to the changing environment. It manages itself and becomes autonomic without human interventions [1]. The four distinct purposes of an autonomic system are: Self-configuring, Self-healing, Self-optimizing and Self-protecting. To achieve those purposes, autonomic systems have a detailed knowledge of their internal state as well as their environment through a continuous monitoring of eventual changes that could affect their components. Detecting changes helps the autonomic system adjust its resources and by monitoring, it continues to determine if the new measures satisfy the desired performance. This is an overview of the closed control loop (Figure 1) of self-management systems.

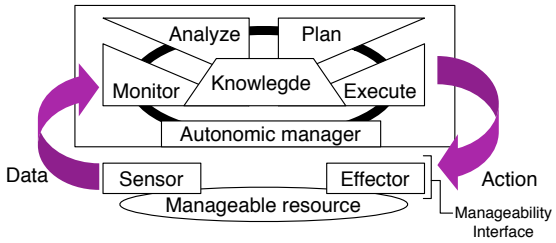


Fig. 1. *The closed control loop for Autonomic computing*

ANA [2] has defined the organization of the autonomic network with the possibility for an autonomic element to exchange information with different neighbor elements. The project FOCALE [3] has defined the ability of an operator to manage an autonomic network with business objective. A new control loop is used to manage the closed control loop of each network element. Because operators don't accept the black box (i.e., closed control loop) solution, an autonomic control loop must be validated by Service/Network Provider before integration in the network.

Self-healing means the system ability to examine, find, diagnose and repair system malfunctions [4]. With the development of autonomic network, the concept of self-healing is becoming more and more important for operators in order to satisfy customers needs. The need for dependable autonomic network has motivated researchers over the recent decades to investigate self-diagnosis. Self-diagnosis is an important part of self-healing. It is required that network could detect faults, handle alarms, execute related test by itself. The purpose of our studies is to introduce self-diagnosis in Self-healing function by autonomic networking.

This paper describes a novel algorithm regarding the use of causal graph model for performing self-diagnosis in telecommunication networks. We have presented the causal graph model in the paper [5]. The causal graph based diagnosis is a kind of model-based approach relied on a graph which represents causal propagation between causes and effects [6]. It could represent a process at a high level of abstraction and could be refined in a very detailed way if necessary. It allows to split the model into small parts and so to have an incremental way of modeling. By the way, it allows a very flexible way to model the knowledge. By definition, the causal graph is an acyclic graph. And it is composed of two parts: nodes representing partial states of the system and arcs representing causal relations. The causal graph structure is based on five types of nodes and one type of arcs (shown in Figure 2):

**Primary cause** is also called default or failure, and has no predecessor in the graph.

**Intermediate cause** represents partial states of the system. Intermediate and initial causes can't be directly observed.

**Observation** is the observable effect of failures and corresponds to alarms. In the graph, these nodes are leaves

 ISBN: 978-1-61208-134-2 

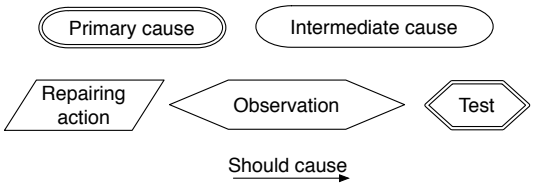


Fig. 2. *The components of causal graph*

and have no successors.

**Test** is used to verify a state or a hypothesis. From the view of causality, they are exactly as observations, the only difference is that they are not automatically notified, but need to be explicitly requested. (This node type could be also used to request a database.)

**Repairing action** is a special node, because it is not a partial state, and it doesnot take part in the causal relation between causes and effects. We can consider it as a label, it only serves when the diagnosis has been completed. These actions are considered as an additional treatment to repair.

**Should cause** means that the cause systematically leads to the effect (or the effect is a systematical effect of the cause). For example, "disconnected Livebox" *should cause* "disconnected IP".

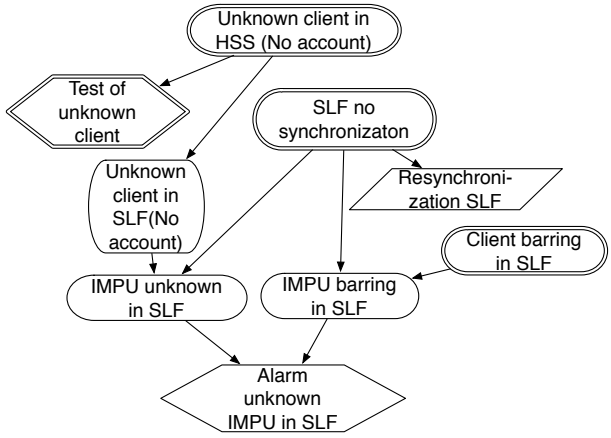


Fig. 3. *An example of a causal graph*

Figure 3 gives an example of causal graph in IMS environment. In this example, the alarm indicates that user's IP Multimedia Public Identity (IMPU) cannot be found in Subscriber Locator Function (SLF). And "Barring" means that a client account exists but has expired. The provisioning of the different databases is not simple because SLF and HSS are duplicated and they need to be updated at the same time with no failure. A non-synchronization failure or an update failure could happen. We refer the reader to [5] for more details. In fact, compared with [5], we have updated, here, some definitions and rules in order to improve and establish the algorithms.

The diagnosis process consists in collecting all the current observations, and then, determining all the primary causes which could explain these observations. Some tests could be launched during diagnosis in order to discriminate remaining ambiguities. Finally, the *repairing action* nodes connect to identify causes which determine the action to execute for repairing. To achieve the process, each node has some associated states:

**Guilty**: for a symptom node (test or observation), it means that the symptom was observed and should be explained by, at least, one of its predecessors. For an intermediate node, it means that the corresponding pathological state is asserted and, for a primary cause, it means that this cause is responsible for the current problem.

**Innocent**: for a symptom node, it means that the symptom is not observed (i.e., an alarm is not present or a test returns a negative result), for a cause primary or not, it means that it is not involved in the causal path and the cause should be searched elsewhere.

The diagnosis algorithms relies on a hypothesis set and are step-by-step procedures. The purpose is to build all the diagnosis by exploring hypothesis.The hypothesis is based on summation of state of causal graph nodes. Meanwhile, the hypothesis must strictly respect the following rules in order to be consistent.

**R1**: if a node is guilty, then all its successors are guilty.

**R1'**: if a node is guilty, at least one of its predecessors should be guilty (except if the node is a primary cause).

**R2**: if a node is innocent, then all its predecessors should be innocent.

**R2'**: if all the predecessors of a node are innocent, this node should be innocent.

**R3**: for a *test* node, its state will be *Guilty* or *Innocent* according to its result of execution (in other words, the execution of a test enables to add the observations in process).

Repairing actions connected to a guilty node of consistent hypothesis are candidates to repair diagnosed problem.

In next section, we will present the definitions needed in order to establish the base of algorithm theories and the formalized diagnosis algorithm.

## II. Definitions and Notations

This section will present required definitions. The model is formally defined as follows.

*Definition 1 (Causal graph):* A causal graph $G$ is an acyclic graph and consists in couple $(\mathbb{N}, S)$, where:

- $\mathbb{N}$ is a set of nodes from the causal graph;
- $S$ is the causal relationship between these nodes.

*Definition 2:* For a causal graph $G = (\mathbb{N}, S)$, and for each node $n \in \mathbb{N}$, we denote:

- $S(n)$: the subset of successors of node $n$;
- $\widehat{S}(n)$: the subset of $\mathbb{N}$ containing all the descendants of $n$ (transitive closure of $S$)
- $S^{-1}(n)$: the subset of nodes whose successor is $n$;
- $\widehat{S}^{-1}(n)$: the subset of all the ascendants of $n$.

As an extension, if $N \subset \mathbb{N}$, we note $\widehat{S}(N)$ the set of all the descendants of nodes $N$:

$$\widehat{S}(N) = \bigcup_{n \in N} \widehat{S}(p) \quad \text{and} \quad \widehat{S}^{-1}(N) = \bigcup_{n \in N} \widehat{S}^{-1}(p)$$

The hypothesis of acyclic graph is then translated by the following property:

*Proposition 1 (Acyclic Graph):* The causal graph $G = (\mathbb{N}, S)$ is an acyclic graph, if and only if

$$\forall n \in \mathbb{N}, n \notin \widehat{S}(n)$$

*Definition 3 (Primary causes and Observations):* For a given causal graph $G = (\mathbb{N}, S)$, we define the set of failures $\mathbb{P}$ and the set of observations $\mathbb{O}$ as follows:

- $\mathbb{P} = \{n \in \mathbb{N} | S^{-1}(n) = \emptyset\}$
- $\mathbb{O} = \{n \in \mathbb{N} | S(n) = \emptyset\}$

Note that $\mathbb{O}$ contains the nodes corresponding to alarms as well as tests.

*Definition 4 (Symptom of a failure):* An observation $a$ ($a \in \mathbb{O}$) is an symptom (e.g., an alarm) of the failure $p \in \mathbb{P}$, iff there is a causal path from $p$ to $a$ (i.e., $a \in \widehat{S}(p)$). The symptoms set of a failure $p$ is thus defined by $\widehat{S}(p) \cap \mathbb{O}$.

*Definition 5 (Independance of failures):* If two failures $p_1$ and $p_2$ are independent, then all the effects (observable symptoms and intermediate causes) of co-occurrence of these two failures is the union of effects of each failure. In other words:

$$\widehat{S}(\{p_1, p_2\}) = \widehat{S}(p_1) \cup \widehat{S}(p_2)$$

*Definition 6 (Diagnosis):* Let a causal graph $G = (\mathbb{N}, S)$ and a set $\mathcal{A} \subset \mathbb{O}$ of alarms (observations), a diagnosis $D$ is a subset of $\mathbb{P}$ which can explain the set of observations. In other words:

$$D \text{ is a diagnosis for } \mathcal{A} \text{ iff : } \quad \mathcal{A} \subseteq \widehat{S}(D) = \bigcup_{p \in D} \widehat{S}(p)$$

Note that if $D$ is a diagnosis for $\mathcal{A}$, then $\forall p \in \mathbb{P}$, if there is no interaction between the failures (hypothesis of the independent failures), $D \cup \{p\}$ is a diagnosis for $\mathcal{A}$.

*Proof:* If $D$ is a diagnosis for $\mathcal{A}$, then we have $\mathcal{A} \subseteq \widehat{S}(D)$. Under the hypotheses of independent failures, we have $\widehat{S}(D \cup \{p\}) = \widehat{S}(D) \cup \widehat{S}(p)$. If $\mathcal{A} \subseteq \widehat{S}(D \cup \{p\})$, then $D \cup \{p\}$ is a diagnosis for $\mathcal{A}$.

For the same set of symptoms $\mathcal{A}$, you can order the corresponding diagnosis using the relation $\subseteq$. We will rather find the minimal diagnosis than the direct diagnosis.

*Definition 7 (Minimal Diagnosis):* Given a causal graph $G = (\mathbb{N}, S))$ and a set $\mathcal{A} \subset \mathbb{O}$ of the alarms (observations), a diagnosis $D$ is minimal diagnosis for $\mathcal{A}$, if and only if, there is no diagnosis $D'$ for $\mathcal{A}$ when $D' \subsetneq D$. In other words, if $D$ and $D'$ are two minimal diagnosis for $\mathcal{A}$ and $D' \subset D$, then $D = D'$.

The objective of diagnosis will be to find all the minimal diagnosis.

*Definition 8 (Hypothesis of diagnosis):* A hypothesis $H$ of diagnosis is defined by:

1) a set $\mathcal{G}(H) \subseteq \mathbb{N}$ of "guilty" nodes;
2) a set $\mathcal{I}(H) \subseteq \mathbb{N}$ of "innocent" nodes;
3) a set $\mathcal{X}(H) \subseteq \mathcal{G}(H)$ containing all unexplained nodes (i.e., without guilty predecessor).

A hypothesis is closed when it explains all its symptoms and remains consistent, that is to say when:

$$\mathcal{X}(H) = \emptyset \quad \text{and} \quad \mathcal{I}(H) \cap \mathcal{G}(H) = \emptyset$$

*Definition 9 (Fusion of hypotheses):* Consider two diagnosis hypotheses $H$ and $H'$, let us define the fusion of these two hypotheses $H \otimes H'$ as follows:

$\mathcal{G}(H \otimes H') = \mathcal{G}(H) \cup \mathcal{G}(H')$

$\mathcal{I}(H \otimes H') = \mathcal{I}(H) \cup \mathcal{I}(H')$

$\mathcal{X}(H \otimes H') = (\mathcal{X}(H) \backslash \mathcal{G}(H')) \cup (\mathcal{X}(H') \backslash \mathcal{G}(H)) \cup (\mathcal{X}(H) \cap \mathcal{X}(H'))$

## III. Diagnosis Algorithms

In this section we propose the base of the implemented diagnosis algorithm and the various proposed extensions to take into account the tests during the current diagnosis. We first introduce a centralized algorithm in this section, and then develop the distributed algorithm in the next section.

### A. Diagnosis Search

The algorithm of calculation $\mathcal{D}$ relies on a set of hypothesis $\mathcal{H}$ and ends when there is no node to explain in the hypotheses (i.e., all hypotheses are closed). During initialization of the algorithm, we start from the initial hypothesis $H_0$ which will be based on $\mathcal{A}$ as all the set of alarms to explain.

**Algorithm 1** Calculation of the set $\mathcal{D}$ of diagnosis

**Require:** $\mathcal{H} = \{H_0\}$ with $H_0$ defined by $\mathcal{G}(H_0) = \mathcal{X}(H_0) = \mathcal{A}$ and $\mathcal{I}(H_0) = \mathbb{O} \setminus \mathcal{A}$

1: **while** $\mathcal{H} \neq \emptyset$ **do**
2: Choose $H$ in $\mathcal{H}$ (and delete it from $\mathcal{H}$)
3: **if** $\mathcal{X}(H) = \emptyset$ (the hypothesis $H$ is closed) **then**
4: The diagnosis $D$ is defined by $D = \mathcal{G}(H) \cap \mathbb{P}$
5: $\mathcal{D} \leftarrow \mathcal{D} \cup \{D\}$
6: **else**
7: Algorithm 2: Develop $H$
8: **end if**
9: **end while**

We can modify the policy of diagnosis progress by changing the procedure of choice $H$ in $\mathcal{H}$ of the diagnosis hypothesis to develop (line 3 of the algorithm 1).

**Algorithm 3** Propagate $m$ in $H'$

1: **if** $\widehat{S}(m) \cap \mathcal{I}(H') \neq \emptyset$ **then**
2: **return** $H'$ is inconsistent
3: **else**
4: $\mathcal{G}(H') \leftarrow \mathcal{G}(H') \cup \widehat{S}(m)$
5: **return** $H'$ is consistent
6: **end if**

 ISBN: 978-1-61208-134-2

**Algorithm 2** Develop $H$ of minimal diagnosis

1: Choose $n \in \mathcal{X}(H)$ (and delete it from $\mathcal{X}(H)$)
2: **if** $S^{-1}(n) = \emptyset$ or $S^{-1}(n) \cap \mathcal{G}(H) \neq \emptyset$ **then**
3: $\mathcal{H} \leftarrow \mathcal{H} \cup \{H\}$ ($n$ is already explained, $H$ remains valid)
4: **else**
5: **for all** $m \in S^{-1}(n)$ **do**
6: Create $H'$ (a duplicate of diagnosis hypothesis $H$)
7: $\mathcal{G}(H') \leftarrow \mathcal{G}(H') \cup \{m\}$
8: $\mathcal{X}(H') \leftarrow \mathcal{X}(H') \cup \{m\}$
9: **execute** Algorithm 3: Propagate $m$ in $H'$
10: **if** H' is consistent **then**
11: $\mathcal{H} \leftarrow \mathcal{H} \cup \{H'\}$
12: **end if**
13: $\mathcal{I}(H) \leftarrow \mathcal{I}(H) \cup \{m\}$
14: **end for**
15: **end if**

Algorithm 2 avoids exploring redundant number of hypothesis and can lead to minimal diagnosis.

*B. Extension of absent alarms or lost alarms*

An absent or a lost alarm corresponds to *an absence of information*, we don't know if the symptom is present or not.

By default, algorithm 1 supposes that if an alarm is unobserved, it is not active (i.e., $\mathcal{I}(H_0) = \mathbb{O} \setminus \mathcal{A}$). If a subset $\mathcal{A}' \subset (\mathbb{O} \setminus \mathcal{A})$ of alarms are absent, then the previous algorithm is initialized with $\mathcal{I}(H_0) = \mathbb{O} \setminus (\mathcal{A} \cup \mathcal{A}')$ and the algorithm remains the same.

The obtained diagnosis assumes that some of these alarms could be active but lost. For a given diagnosis $D$, the set of active alarms is: $\bigcup_{p \in D} \widehat{S}(p) \cap \mathbb{O}$. And so, the alarms assumed as lost by this same diagnosis are: $\left( \bigcup_{p \in D} \widehat{S}(p) \cap \mathbb{O} \right) \setminus \mathcal{A}$.

*C. Extension of tests*

We want to handle the case where certain observations don't show spontaneously but correspond to tests to be executed. In this case, one of certain observations $\mathbb{O}$ are labeled as tests (belongs to $\mathbb{T}$).

The algorithm 4 is the main body of the algorithm, and it will rely on the following procedures: Algorithm 5 and Algorithm 6. Note that, for a test node, the algorithm 6 will replace the algorithm 3 to execute.

In the algorithms, $\mathcal{T}^*$ means the test before the execution or the test waiting for the execution. $\mathcal{T}^+$ and $\mathcal{T}^-$ are the results after test execution, here $\mathcal{T}^-$ stands for a positive result (Innocent) and $\mathcal{T}^+$ for a negative one (Guilty).

The test choice policy (line 1 of the algorithm 5) could be based on the cost induced by the test (in terms of delay, of computer resource, etc. ).

The development of an algorithm can go through the selection of executed tests.

**Algorithm 4** Calculation of the set $\mathcal{D}$ of diagnosis

**Require:** $\mathcal{A} \cup \mathcal{A}' = \emptyset$
1: $\mathcal{T}^* = \mathcal{T}^+ = \mathcal{T}^- = \emptyset$
2: $\mathcal{H} = \{H_0\}$ with $H_0$ defined by $\mathcal{G}(H_0) = \mathcal{X}(H_0) = \mathcal{A}$ and $\mathcal{I}(H_0) = \mathcal{A}'$
3: **while** $\mathcal{H} \neq \emptyset$ **do**
4: Choose $H$ in $\mathcal{H}$ (and delete it from $\mathcal{H}$)
5: **if** $\mathcal{G}(H) \cap \mathcal{T}^- \neq \emptyset$ ou $\mathcal{I}(H) \cap \mathcal{T}^+ \neq \emptyset$ **then**
6: $H$ is consistent (and abandoned as in contradiction with tests executed in other hypothesis)
7: **else**
8: **if** $\mathcal{X}(H) = \emptyset$ **then**
9: **if** $\mathcal{T}^* \cap \mathcal{G}(H) = \emptyset$ **then**
10: (the hypothesis $H$ is terminated and without waiting)
11: The diagnosis $D$ is defined by $D = \mathcal{G}(H) \cap \mathbb{P}$
12: $\mathcal{D} \leftarrow \mathcal{D} \cup \{D\}$
13: **else**
14: **execute** Algorithm 5: Launch a test for $H$
15: **end if**
16: **end if**
17: **else**
18: **if** $\mathcal{T}^* \cap \mathcal{G}(H) = \emptyset$ (No tests for $H$) **then**
19: **execute** Algorithm 2: Develop $H$
20: **else if** We choose to launch a test for $H$ **then**
21: **execute** Algorithm 5: Launch a test for $H$
22: **else**
23: **execute** Algorithm 2: Develop $H$
24: **end if**
25: **end if**
26: **end while**

**Algorithm 5** Launch a test for $H$

1: Choose and execute a test $t \in \mathcal{T}^* \cap \mathcal{G}(H)$
2: **if** The result of $t$ is a "detected problem" **then**
3: $\mathcal{T}^+ \leftarrow \mathcal{T}^+ \cup \{t\}$
4: $\mathcal{H} \leftarrow \mathcal{H} \cup \{H\}$ ($H$ is consistent)
5: **else**
6: $\mathcal{T}^- \leftarrow \mathcal{T}^- \cup \{t\}$
7: (we abandon $H$ which is in contradiction with the result of test)
8: **end if**

## IV. Distribution of the algorithm

A distributed system is a set of components that interact by exchanging messages through a communication network. When detecting an anomaly, components emit alarms for monitoring the system.

For a distributed architecture: each local diagnosis system first calculates a diagnosis using only local information, then a global diagnosis is determined by communication between each local diagnoser. This approach has the advantage of a complete distribution of the diagnosis task. It seems also interesting to make the diagnosis as close as

 ISBN: 978-1-61208-134-2

**Algorithm 6** Propagate $m$ in $H'$

**if** $\widehat{S}(m) \cap \left(\mathcal{I}(H') \cup \mathcal{T}^-\right) \neq \emptyset$ **then**
  **return** $H'$ is inconsistent
**else**
  $\mathcal{G}(H') \leftarrow \mathcal{G}(H') \cup \widehat{S}(m)$
  $\mathcal{T}^* \leftarrow \mathcal{T}^* \cup \left(\left(\widehat{S}(m) \cap \mathbb{T}\right) \setminus \mathcal{T}^+\right)$
  **return** $H'$ is consistent
**end if**

possible to the source of alarms, because it causes fewer problems with delays and lost alarms. Nevertheless there is a risk of generating a large number of communications. Some approaches focus on issues related to distributed algorithms, and propose distributed monitoring methods based on several communicating supervisors [7][8][9]. Other approaches combine distributed algorithms with the characterization of system behaviors [10][11][12][13][14]. The present algorithms belong to this second category.

### A. The principle

The distributed algorithm for the autonomic networks is based on network architecture: the different nodes of causal graph correspond to various components states of autonomic systems (network, IMS component, etc. ).

The principle is to cut the causal graph according to components of system architecture. Each autonomic component (or group of components) has its own diagnoser that will establish its diagnosis hypotheses according to its own causal graph (subgraph of the global causal graph) and will compare to the other neighbor diagnosers in order to get a compatible and consistent diagnosis.

The availability of distributed algorithm could be assured by the two following points:

- the local diagnosis is compatible with the local observations and the local causal subgraph.
- all the local diagnosis should be compatible with at least one local diagnosis of all its neighbours.

### B. Algorithm

The local diagnosis is based on a subgraph $G' = (\mathbb{N}', S)$ of $G = (\mathbb{N}, S)$, where the set $\mathbb{N}' \subset \mathbb{N}$ and the relation $S$ applied on $\mathbb{N}'$. The symptoms of the local diagnoser are: $\mathcal{A} \cap \mathbb{N}'$.

Some nodes of $\mathbb{N}'$ are connected with the other nodes which aren't in $\mathbb{N}'$ (not in this local graph).These nodes are called "connection nodes" between the local diagnosers. The change of diagnosis hypotheses of these "connection nodes" will trigger message exchange between diagnosers.

*Definition 10 (Boundary of effects and of causes):* Given a causal graph $G = (\mathbb{N}, S)$ and a sub-graph $G' = (\mathbb{N}', S)$ de $G$, where $\mathbb{N}' \subset \mathbb{N}$.
The boundary of effects in $G'$ is defined by:

$$\mathbb{F}_{G'} = \left\{ n \in \mathbb{N} \middle| n \notin \mathbb{N}' \text{ et } \left(S^{-1}(n) \cup \mathbb{N}'\right) \neq \emptyset \right\}$$

The boundary of causes in $G'$ is defined by:

$$\mathbb{F}_{G'}^{-1} = \left\{ n \in \mathbb{N}' \middle| S^{-1}(n) \not\subset \mathbb{N}' \right\}$$

We assume the network architecture is hierarchical, which enable be share the task of diagnosis between different levels of abstraction of a given system.

### Distributed algorithm

Assume a partition of $\mathbb{N} = \mathbb{N}_1 \cup \cdots \cup \mathbb{N}_N$, and so a set of local subgraph $G_i = (\mathbb{N}_i, S)$, $\forall i, j,\ G_i \cap G_j = \emptyset,\ 1 \leqslant i \leqslant N$ and $1 \leqslant j \leqslant N$. The distributed algorithm is based on messages exchanged between neighbor diagnosers: each subgraph sends and updates a message $\mathcal{M}$ ($\mathcal{M} = \mathcal{H}$) for each neighbor. The idea is that: when $G_i$ receives $\mathcal{M}$ from its neighbor $G_j$, then $G_i$ fuses $\mathcal{H}_i$ with $\mathcal{H}_j$ (if two hypotheses are coherent), and then $G_i$ will recalculate $\mathcal{D}$ to update it; if $\mathcal{D}$ is complete, $G_i$ sends result of $\mathcal{D}$ to operator, creates at the same time an unique $ID$ and sends a message $askE_{ID}$ to ask if its neighbors have finished their diagnosis. Related algorithms (Algorithms 7, 8 and 9) are the following:

**Algorithm 7** Fusion of $\mathcal{H}_i$ and $\mathcal{H}_j$

**for all** $(H_i, H_j) \in \mathcal{H}_i \times \mathcal{H}_j$ **do**
  **if** $H_i \otimes H_j$ is coherent **then**
    $\mathcal{H}_i' \leftarrow \mathcal{H}_i' \cup \{H_i \otimes H_j\}$
  **end if**
**end for**
$\mathcal{H}_i \leftarrow \mathcal{H}_i'$
**execute** Algorithm 8: Diagnosis $\mathcal{D}$ in subgraph

**Algorithm 8** Diagnosis $\mathcal{D}$ in subgraph

**Require:** $\mathcal{M} \leftarrow \emptyset$
**while** $\mathcal{H} \neq \emptyset$ **do**
  Choose $H$ in $\mathcal{H}$ (and delete it from $\mathcal{H}$)
  **if** $\mathcal{X}(H) = \emptyset$ **then**
    sends $H$ to operator
  **else if** $\mathcal{X}(H) \cap \mathbb{F}_{G'}^{-1} \neq \emptyset$ **then**
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{H\}$
  **else**
    **execute** Algorithm 2: Develop $H$
  **end if**
**end while**

**Algorithm 9** Termination of $\mathcal{D}$

**if** $\mathcal{M} = \emptyset$ **then**
  Creat an unique $ID$
  sends $askE_{ID}$ to its neighbor diagnoser
**else**
  sends $\mathcal{M}$ to its neighbor diagnoser
**end if**

 ISBN: 978-1-61208-134-2 

In these 3 algorithms, we have not yet add the *test* and *repairing actions* in order to simplify the algorithms.

**Termination of algorithm**

For a distribution of algorithm, the termination of diagnosis should be considered. When $\mathcal{H} = \emptyset$, hypothesis of diagnosis is closed (i.e., diagnosis is complete). Then, all the results of diagnosis will be sent to operator to finally validate and execute the repairing actions.

Let us define a special message $okE_{ID}$ as the response of $askE_{ID}$ to notice the agreement of "complete diagnosis" between the neighbor diagnosers and a message $cancelE_{ID}$ as a notice of deleting the finished diagnosis.

**Algorithm 10** Reception $askE_{ID}$ from $G_j$

1: **if** $H \neq \emptyset$ **then**
2: sends $H$ to neighbor diagnosers
3: delete $H$ from $G_i$
4: **else if** $E_{ID} \in \mathcal{E}$ **then**
5: sends $okE_{ID}$ to $G_j$
6: **else**
7: $\mathcal{E} \leftarrow \mathcal{E} \cup \{E_{ID}\}$
8: $\mathcal{W}(E_{ID}) \leftarrow$ {all the $ID$s of neighbor diagnosers except $G_j$}
9: $\mathcal{S}(E_{ID}) \leftarrow G_j$
10: sends $okE_{ID}$ to all $\mathcal{W}(E_{ID})$
11: **end if**

**Algorithm 11** Reception $okE_{ID}$ from $G_j$

1: $\mathcal{W}(E_{ID}) \leftarrow \mathcal{W}(E_{ID}) \setminus \{G_j\}$
2: **if** $\mathcal{W}(E_{ID}) = \emptyset$ **then**
3: sends $okE_{ID}$ to $\mathcal{S}(E_{ID})$
4: **end if**

**Algorithm 12** Reception $cancelE_{ID}$ from $G_j$

1: sends $cancelE_{ID}$ to all $\mathcal{W}(E_{ID})$
2: $\mathcal{W}(E_{ID}) \leftarrow \emptyset$

In the algorithm 10, $\mathcal{E}$ is the set of $ID$ of all the neighbors which send $askE_{ID}$. $\mathcal{W}(E_{ID})$ is the set of all the neighbors which have sent $askE_{ID}$ and wait for the response $okE_{ID}$. And $\mathcal{S}(E_{ID})$ is the source of diagnosis.

## V. Conclusion and Future Work

We have introduced a novel algorithm regarding the use of causal graph model for performing self-diagnosis. To do so, we first introduced the notion of causal graph modeling. Mathematically, it's possible to derive diagnosis algorithm based on causal graph: local diagnosis are executed by local diagnosers and take the form of message passing algorithms. The keystone of the algorithms is to achieve a self-diagnosis in a global environment.

Future work will focus on the implementation of the self-diagnosis algorithm in an IMS platform to verify diagnosis performance, and then, according to the results of diagnosis, we will try to refine the policies of the algorithms and to improve it.

## References


[1] IBM, *Autonomic Computing: IBM's perspective on the state of information technology*, Technical report, 2001.

[2] ANA: Autonomic Network Architecture, http://www.ana-project.org

[3] J. C. Strassner, N. Agoulmine, and E. Lehtihet, *FOCALE - A novel autonomic networking architecture*. pp.64-79, 2007.

[4] D. Tosi, *Research Perspectives in Self-healing Systems*, Technical Report of the University of Milano-Bicocca. July, 2004.

[5] J. Lu, C. Dousson, B. Radier, and F. Krief, *Towards an autonomic network architecture for self-healing in telecommunications networks*, in Proceedings of 4th International Conference on AIMS. Zurich, Switzerland, 2010.

[6] L. Console, D.T. Dupre, and P. Torasso, *A theory of diagnosis for incomplete causal models*, in Proceedings of IJCAI. USA, 1989.

[7] R. K. Boel and J.H. van Schuppen, *Decentralized Failure Diagnosis for Discrete Event Systems with Costly Communication between Diagnosers*, in proc. 6th Int. Workshop on Discrete Event Systems. WODES' 02, pp.175-181, 2002.

[8] R. Debouk, S. Lafortune, and D. Teneketzis, *Coordinated decentralized protocols for failure diagnosis of discrete event systems*, Discrete Event Dynamic Systems : theory and applications, vol. 10(1/2), pp. 33-86, 2000.

[9] T. Yoo and S. Lafortune, *A General Architecture for Decentralized Supervisory Control of Discrete-Event Systems*, Discrete Event Dynamic Systems: Theory and Applications. vol. 12(3), pp. 335-377, July, 2002.

[10] R. Su, W.M. Wonham, J. Kurien, and X. Koutsoukos, *Distributed Diagnosis for Qualitative Systems*, in proc. 6th Int. Workshop on Discrete Event Systems, WODES'02, pp. 169-174, 2002.

[11] Y. Pencole, M. Cordier, and L. Roze, *A decentralized model-based diagnostic tool for complex systems*, Int. J. on Artif. Intel. Tools. World Scientific Publishing Comp., vol. 11(3), pp. 327-346, 2002.

[12] S. Genc and S. Lafortune, *Distributed Diagnosis Of Discrete-Event Systems Using Petri Nets*, in proc. 24th Int. Conf. on Applications and Theory of Petri Nets. LNCS 2679, pp. 316-336, June, 2003.

[13] R. K. Boel and G. Jiroveanu, *Distributed Contextual Diagnosis for very Large Systems*, in Proceedings of WODES'04, pp. 343-348, 2004.

[14] E. Fabre, A. Benveniste, S. Haar, and C. Jard, *Distributed Monitoring of Concurrent and Asynchronous Systems*, Journal of Discrete Event Systems, 2005.


 ISBN: 978-1-61208-134-2 