ORIGINAL RESEARCH

A Bayesian Network approach to diagnosing the root cause of failure from trouble tickets

Fco. Javier Molinero Velasco

Telefónica Investigación y Desarrollo, Madrid, Spain

Correspondence: Fco. Javier Molinero Velasco. Address: Telefónica Investigación y Desarrollo, Distrito C – Edificio Oeste 1 – 5ª planta, Ronda de la Comunicación s/n, 28050 Madrid, Spain. Email: fjmv@tid.es.

Received: May 7, 2012 **Accepted:** July 24, 2012 **Published:** December 1, 2012
DOI: 10.5430/air.v1n2p75 **URL:** http://dx.doi.org/10.5430/air.v1n2p75

Abstract

Telecommunications networks comprise elements of very different types that work together to provide services. Quite often, hardware failures are interrelated and it is hard for technicians specialized in specific hardware to find out these relationships. In this context, Bayesian Networks (BN) provide a good and flexible solution because they allow us to model the causal relationships between element failures and infer information from existing evidence. The goal is that network technicians can be informed of the real scope of failures and the probable existence of root problems, thus optimizing resources and reducing recovery time. Besides, with this approach a real element hierarchy can be built, allowing the discovery of hidden dependencies between elements. The outcome of this work has been the development of a rooting module attached to an incident management system (trouble ticketing system, TT).

Key words

Bayesian, Network, Incident, Root, Graph, Trouble ticketing

1 Introduction

Telecommunication networks consist of thousands of different hardware elements of the most varied kinds: servers, routers, modems, switching units, cables, base stations, cooling elements, energy elements, etc. Many of the possible relations between elements are not explicitly defined, even more when they are heterogeneous. For example, there is a clear relation between the cooling system that controls a room temperature and all the hardware installed in the room; a failure in the former will probably affect a smooth running of the hardware or make it break down. Another typical case is a fibber cut-off that makes many dependent mobile base stations be in turn cut off, consequently affecting many customers. While in certain types of hardware elements this information is explicitly modelled, it is not the case in many others. This makes it difficult to use traditional programming solutions to automatically link failures to a root cause [1]. Incident management systems, usually known as trouble ticketing (TT) systems, offer the technician the possibility to link an incident produced on an element to another existing incident, creating a child-parent relation. So, it depends on expert knowledge to be able to identify these situations quickly. Many times it is not until several similar incidents have appeared and technicians have dedicated much time and resources to analyze them all that a root cause is discovered to be the real problem. In the mean time many customers could have their services partially or fully affected. What makes matters worse is the fact that it is precisely root causes that are usually harder to detect and affect more customers. The inverse way is also

important, i.e. predicting how a failure in an element can affect other elements, thus being able to evaluate the real scope of the problem as soon as possible [2, 3].

Different AI approaches are applied to address a variety of problems in the Telco area, mostly churning detection [5], and forecasting [6]. However, incident management systems have hardly used AI techniques to optimize the processes involved. A lot of work has been dedicated to supervision in order to prevent or detect problems as soon as possible, but little to nothing has been done once the incident has been created. In most cases the optimizations are reduced to more or less sophisticated decision rules, or searching for previous similar cases in knowledge bases in what is known as case based reasoning [4]. These approaches may be suitable for simple problems but are not adequate to address complex or changing environments. In recent years machine learning techniques are starting to be applied in the context of TT systems to discover information and automate certain tasks [7-9].

An example of complexity is the case of TT systems that manage incidents from large heterogeneous networks under difficult conditions: an incident could affect an important service offered to hundreds of people, thousands of incidents may appear every day, and the topology of the network is complex. Under these conditions, decisions cannot be delayed and actions must be carried out right away.

In recent years, with the development of efficient computational algorithms, Bayesian networks have had a revival within the AI community. BN's causal semantics allows the representation of causal relationships between the variables. BNs model the quantitative strength of the connections between variables, allowing probabilistic beliefs about them to be updated automatically as new information becomes available. This allows inference and reasoning under uncertainty, probabilistically, in what is called Bayesian reasoning. As BNs provide full representations of probability distributions over their variables, they can be conditioned upon any subset of them, supporting any direction of reasoning. For example, diagnostic reasoning, that goes from symptoms observed to causes; or predictive reasoning, that goes from new information about causes to new beliefs about effects [10, 11].

All this makes BNs a good AI technique to address the problem of finding the root cause of an incident described in this article [12, 13].

The aim of this work is keeping a real time directed graph of interconnected elements, where each node indicates the current probability of that element having an incident, and where an edge going from an element A to an element B shows the current probability of a failure in A being the cause of a failure in B.

This article describes how Bayesian networks combined with classification algorithms can be used in the scope of telecommunications networks to address the aforementioned problems. Being this a practical work that is to be applied to a real TT system with several thousand incidents created daily, the solution pays important attention to performance issues, considering that response time is critical (it is no use providing very good results once incidents have already been solved).

The article is structured as follows: the context and overall description is presented in Section 2. A detailed description showing the application of Bayesian Networks is presented in Section 3. Section 4 describes how to integrate a rooting module into a real TT system. Then, Section 5 presents a proof of concept based on this work applied to a real TT system and the results obtained. Finally, Section 5 offers the conclusions.

2 Overall description

The data for this work are incidents produced on network elements. Every incident describes a problem detected on a specific element, and contains information such as the type of element, location, service, type of symptom, and problem

 ISSN 1927-6974 E-ISSN 1927-6982

description. Sometimes, a failure detected on an element has really been caused by another element it depends on. When technicians discover such a situation, they create a second incident (in case it does not already exist) and link the first incident to this second one, establishing a parent-child relationship. Due to the elements heterogeneity, this dependency relation is not explicitly defined in any inventory system, so that it is technicians' expert knowledge and analysis that guide them to create this relation between incidents. In some cases a failure in an important element can produce a cascade of failures in many other elements. What technicians see are a lot of different unrelated incidents, so that it is hard for them to evaluate the real situation to be able to link all incidents to a root incident. There exist different situations: the root element failure is detected first, and so its incident is created first; or the affected elements failures are detected first. In any case, in more or less degree, the situation is difficult to manage. When the elements belong to different types of networks, incidents are assigned to different groups of technicians, making it even more difficult for them to realize what the real situation is.

This article describes the work done so far in order to help technicians to manage these situations. The basic idea consists on automatically learning the relation between elements with no need of an inventory system, by just watching what has happened in the past, how elements related to each other either by means of explicitly linked incidents or through time association of incidents occurring simultaneously. This learning process is applied to the massive information accumulated through years to build conditional probability tables (CPTs) that express the probability of a failure in an element being the root cause of a failure in another element. These initial CPTs are afterwards updated with current information on a regular basis.

CPTs are used to build different temporal BNs for groups of related elements. Based on the posterior probabilities obtained from these temporal BNs we can represent a real time directed graph of interconnected network elements. This graph represents the probabilities or believes of elements having a failure, and the probabilities or believes of relations between elements in terms of failures. Therefore, the graph holds the complete information of the state of the network at any given time. Figure 1. Probability Graphshows an example graph with nine elements E_i, i=1, 2, … 9. Each node Pi represents the current probability of a failure in element E_i, and each edge P_{ij} represents the current probability of E_i being the root cause of E_j.

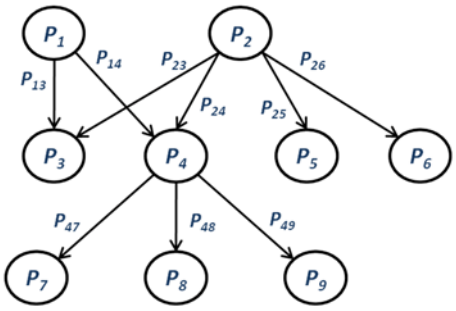

Figure 1. Probability Graph

In our case, this graph has around 100,000 nodes and 50,000 edges. This graph is not static, it is periodically updated as problems are created, solved, or linked. These three possible types of events are introduced as new evidence into the different BNs, and posterior probabilities are automatically adjusted, so that we get a real time snapshot of the probability of failure of all existing elements in the network. Furthermore, we are offered an explanation of failures through the posterior probabilities associated to the links between elements.

The real benefit of this approach is that the relevant part of that graph can be attached to incidents as they are created, offering information about possible affected elements, and possible root causes. Moreover, different rules can be defined and applied to warn technicians of different situations: an element whose failure probability has increased over a given

threshold => probable new incident, and an edge whose probability has increased over a given threshold => probable new link

Real Time Graph (see Figure 2) shows an example of how a possible real time situation is reflected in the graph. A possible sequence of events could have been this:

- t_0: Initially there are no incidents. The graph shows the prior probabilities.
- t_1: An incident is detected on element E_2. This evidence is inserted into the related BNs and the resulting graph is updated. As a result, P_2 is set to 1 and the rest of the links and elements have their probabilities increased.
- t_2: A new incident is detected, this time on E_4. This second evidence is processed and the graph is consequently updated. Consequently, P_4 is set to 1, and P_3 and P_1 as well as the corresponding edges have their probabilities increased once more.

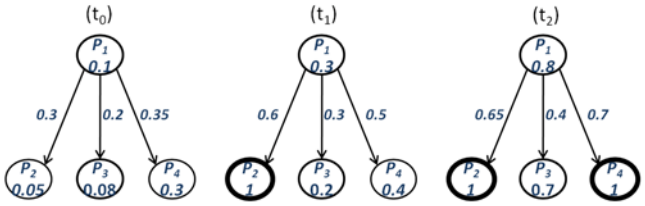

Figure 2. Real Time Graph

The final situation shows that the probable root cause of incidents on E_2 and E_4 is a failure in E_1. It also shows that E_3 could also be affected.

3 Detailed description

The final goals of this work are two: given a problem, predict its root cause, and predict its real scope. To achieve these goals a directed graph like the one shown in Figure 1. Probability Graph, is built. This graph describes all possible relations between elements, and offers two types of real time information: the probability of an element having a problem, and the probability of an element causing a problem in another element.

The procedure to build and keep this graph updated consists of several steps: (i) defining the graph structure; (ii) designing a specific Bayesian network structure or template in order to express the elements relations and calculate the graph probabilities; (iii) automatically learning relations from past experience in order to build the necessary conditional probability tables of elements relationships; (iv) rebuilding and updating evidence in Bayesian networks as new incidents are created, closed, and linked, in order to keep the graph updated.

The description of the graph structure, the Bayesian Network, the construction of the Conditional Probability Tables, and the management of the graph are described in the subsections below.

3.1 Graph structure

The graph structure is built from historical data in the following way: there is a node in the graph for every element Ei in the universe U={E1, ..EN}; if there has ever been a problem in element Ei that has been caused by a problem in element Ej, then there is an arrow from Ej to Ei, indicating a parent->child relationship; each node represents the current probability of a failure on its associated element; each arrow represents the probability of the parent node being the cause of the problem on the child node. The calculation of probabilities is described in later sections.

 ISSN 1927-6974 E-ISSN 1927-6982

3.2 Bayesian network structure

Every element E_i in the graph has its own Bayesian network B_{Ni}. This BN has a node for Ei (the root element, denominated R) and nodes for all its children, children (E_i), with the structure shown in figure 3.

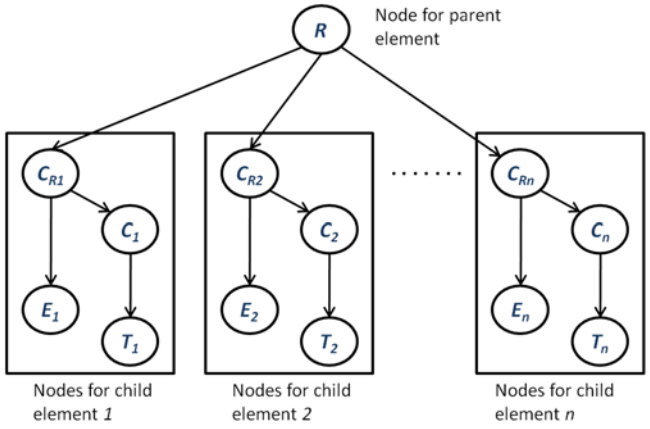

Figure 3. Bayesian Network Structure

It has the following type of nodes:

R: represents the probability of element R having a problem.

C_{Ri}: represents the probability of R being the root cause of a problem in E_i.

E_i: represents the probability of element E_i having a problem.

C_i: represents the probability of element E_i having a problem that is caused by some other existing problem.

T_i: represents the result of a test on the existence of a root cause for E_i having a problem [2].

This structure fulfills the following requirements: it expresses the relations and dependencies between elements; its conditional probabilities can be calculated from existing historical data; it incorporates testing nodes to be able to express the result of predictions; and it is simple enough so that all BNs can be built and kept in a real time basis.

Next section shows how the probability tables for all types of nodes can be calculated.

3.3 Conditional probability tables

The conditional probability tables of the different types of nodes are calculated from historical data in the following way:

- Root node R: P(R=T) or prior probability that R is failing can be calculated dividing the total time that element R has failed ($T_{R\ failure}$) by the total time that R has existed ($T_{R\ total\ time}$).

$$P(R = T) = \frac{T_{R\ failure}}{T_{R\ total\ time}} \quad (1)$$

- Nodes CRi

$P(C_{Ri}=T|R=T)$ or probability that element E_i is failing being R its root cause, conditioned to "R is failing" can be calculated in the following way:

$$P(C_{Ri}=T|R=T)=P(C_{Ri}=T,E_i=T|R=T)+P(C_{Ri}=T,E_i=F|R=T)$$

$$=P(C_{Ri}=T,E_i=T|R=T)=P(C_{Ri}=T|R=T,E_i=T)P(E_i=T|R=T)$$

$$=\frac{N_{Ri}}{N_{R\ and\ i}}*\frac{N_{R\ and\ i}}{N_R}=\frac{N_{Ri}}{N_R} \quad (2)$$

Where:

N_{Ri}: is the number of times that element R has been the root cause of a failure in Ei.

$N_{R\ and\ i}$: is the number of times that elements R and Ei have failed simultaneously.

N_R: is the number of times that element R has failed.

P ($C_{Ri}=T|R=F$) or probability that element E_i is failing being R its root cause, conditioned to "R is not failing" is 0.

$$P(C_{Ri}=T|R=F)=0 \quad (3)$$

- Nodes E_i:

$P(E_i=T|C_{Ri}=T)$ or probability that element E_i is failing conditioned to R being the root cause of a failure in E_i is 1.

$$P(E_i=T|C_{Ri}=T)=1 \quad (4)$$

$P(E_i=T|C_{Ri}=F)$ or probability that element E_i is failing conditioned to R not being the root cause of a failure in E_i:

$$P(E_i=T|C_{Ri}=F)=\frac{P(C_{Ri}=F|E_i=T)P(E_i=T)}{P(C_{Ri}=F)}$$

$$=\frac{P(C_{Ri}=F|E_i=T)P(E_i=T)}{P(C_{Ri}=F|E_i=T)P(E_i=T)+P(C_{Ri}=F|E_i=F)P(E_i=F)}$$

$$=\frac{P(C_{Ri}=F|E_i=T)P(E_i=T)}{P(C_{Ri}=F|E_i=T)P(E_i=T)+P(E_i=F)} \quad (5)$$

- Nodes C_i:

$P(C_i=T|C_{Ri}=T)$ or probability that Ei has a failure that is caused by some root cause, conditioned to R being the root cause of a failure in E_i is 1.

$$P(C_i=T|C_{Ri}=T)=1 \quad (6)$$

P(Ci=T|$C_{Ri} = F$) or probability that Ei has a failure that is caused by some root cause, conditioned to R not being the root cause of a failure in Ei:

$$P(C_i = T|C_{Ri} = F) = \frac{P(C_{Ri} = F|C_i = T)P(C_i = T)}{1 - P(C_{Ri} = T)}$$

$$= \frac{P(C_{Ri} = F|C_i = T)[P(C_i = T|E_i = T)P(E_i = T) + P(C_i = T|E_i = F)P(E_i = F)]}{1 - [P(C_{Ri} = T|E_i = T)P(E_i = T) + P(C_{Ri} = T|E_i = F)P(E_i = F)]}$$

$$= \frac{P(C_{Ri} = F|C_i = T)P(C_i = T|E_i = T)P(E_i = T)}{1 - P(C_{Ri} = T|E_i = T)P(E_i = T)} \quad (7)$$

- Nodes Ti: This node assumes the existence of a model to predict whether an existing failure on an element is being caused by some other element. P(Ti=T|$C_i = T$) and P(Ti=T|$C_i = F$) are given by true positives and false positives frequencies of the algorithm, respectively.

The different conditional probabilities in the equations can be calculated from historical data:

$$P(E_i = T) = \frac{T_{i\,failure}}{T_{i\,total\,time}} \quad (8)$$

$$P(C_{Ri} = T|C_i = T) = \frac{N_{Ri}}{N_{Ci}}$$

$$P(C_{Ri} = T|E_i = T) = \frac{N_{Ri}}{N_i} \quad (9)$$

Where:

N_{Ri}: is the number of times that element R has been the root cause of a failure in E_i.

N_i is the number of times that element Ei has failed.

Nc_i is the number of times that a failure in E_i has been caused by some other element.

As indicated in the previous section, a prediction model is used to calculate the CPTs of nodes T_i. To build this model different information about a ticket can be used: type of element, symptoms, severity, etc. Some of this information can be textual, filled in by a supervisor or an automatic supervision system, so that a text mining process must be performed.

The type of model to use is not relevant in this work, although different models have been tested with good results, such as naive Bayes and decision trees. As a result of the model we can get a transition matrix and use the true positives and false negatives frequencies as parameters for the CPTs. There could be one model for all type of elements, in which case all T_i nodes would have the same parameters, or a different model for each type of element, in which case there would be different sets of parameters, one for each type of element.

Once the model is generated we can apply it to new tickets to predict whether they have a root cause, which translates into True or False evidence in nodes T_i.

3.4 Graph management

Once the BNs templates and the way to calculate the CPT are defined the complete graph can be constructed as follows:

- The nodes and arrows are defined as described in 3.1.
- Initially all P_i and P_{ij} are initialized to very low values.
- Periodically the following procedure is started:

  For every node Pi in the graph, representing element E_i:

  - B_{Ni} is created as described in 3.2
  - All the necessary CPTs are calculated as described in 3.3
  - All existing evidence is added to B_{Ni}. The possible evidences are:
    - Failure in the root element R
    - Failure in child E_j
    - Evidence of E_i (node R) being the root cause of a failure on child E_j, C_{Rj}
    - Evidence of a child E_j being linked to some root cause other than E_i (node R), C_{Rj}
    - Evidence from tests T_j: in case that a child is failing, a test can be applied to predict whether there is some external root cause
  - Posterior probabilities obtained from B_{Ni} are updated into the graph:
    - Nodes
      - If $P(Ei|evidence)>P_i$ then $P_i = P(Ei|evidence)$
      - If $P(Ei|evidence)<P_i$ and P_i was assigned by this same B_{Ni} in some previous calculation process then $P_i = P(E_i|evidence)$
    - Arrows
      - If $P(C_{Rj}|evidence)>P_{ij}$ then $P_{ij}= P(C_{Rj}|evidence)$
      - If $P(C_{Rj}|evidence)<P_{ij}$ and P_{ij} was assigned by this same B_{Ni} in some previous calculation process then $P_{ij} = P(C_{Rj}|evidence)$

4 Integration with trouble ticketing systems

The procedure described earlier can easily be integrated into a TT system, i.e., a system that manages incidents or failures on elements. Concretely, a trouble ticketing system that manages incidents produced in the infrastructure of a telecommunications company. In this scenario there are thousands of incidents every day, covering many different types of elements. All network elements collaborate in daily operations, providing all the necessary services requested by customers. For example, in order for a mobile call to work a complete series of hardware and software elements must work, energy systems must be operative, and cooling systems must regulate adequately room temperatures. All the elements must work simultaneously, and any failure in this chain can make the call to fail.

The typical flow of events in TT systems is as follows: a supervision system detects an anomaly and issues an alarm. When this happens a new ticket is created in the TT system, indicating the anomaly detected on the element. The ticket is automatically assigned to a group of specialized technicians who analyse the ticket and try to solve the problem. Many times they find out that what is making the element fail is a problem in some other network element, in which case they create a new ticket (root or parent ticket) on this other element (if it does not already exist), and link both tickets, creating a parent-child relation.

 ISSN 1927-6974 E-ISSN 1927-6982

A system based on the procedures described in the previous sections, that we can call a Rooting System (RS), can operate as a separate and independent module from the TT system to help automate the creation of parent-child relations. The RS has access to all historical data in order to build the probability tables, and receives real time events from the TT system: ticket creation, closing, and new links. New events make the system learn from the environment, allowing it to update the probability tables of all affected BNs. Besides, new evidence triggers the Bayesian reasoning process which in turn updates the state of the graph as described above.

Rooting System Integration (see Figure 4) shows how the RS integrates with the TT system. The RS offers different degrees of interaction with the TT system. Technicians can view or search the probability graph to get relevant information about the predictions on the state of the network. The RS issues alarms when probabilities go beyond specified thresholds indicating different conditions. Possible alarms are: (i) a ticket should be linked to another existing root ticket; (ii) a new ticket should be created, with other existing tickets being linked to it; (iii) an existing ticket can affect other elements which still don't have tickets. Finally, the system offers information about the real scope of a problem. In some of these situations, the RM could automatically act, creating or linking tickets.

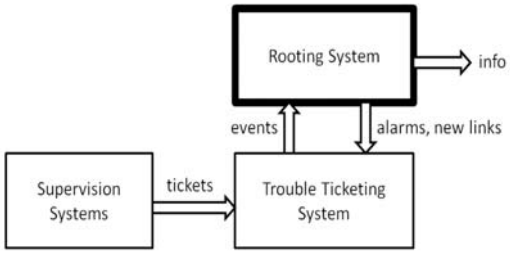

Figure 4. Rooting System Integration

5 Results

A proof of concept applying the methods described in this article has been applied and tested in a real TT system that manages around five thousand tickets every day. The trouble tickets describe problems with switching, transmission, mobile, energy, servers, and cooling elements. Tickets include information about the affected element (identifier, type, location, services offered …), failure information (severity, affected clients, symptoms …), and management information (ticket identifier, technician in charge, dates, tasks performed, parent ticket identifier …). In order to build the different CPTs of BNs, historical data were processed. Information about parent-child relations as well as ticket opening and closing dates where used to obtain CPTs for all types of nodes except Ti nodes, as detailed in section 3.3.

For T_i nodes a Random Forest (RF) model was built using the type of element, initial severity, problem description, and type of symptom, as described in section 3.4. Most of the tickets are created automatically by supervision systems, and there is a specialized supervision system for each type of network. Therefore, because the problem description included in each ticket is specific to its type of network, in order to improve the results, a different RF model was trained for each type of network element. The true positives vary between 78% (energy) and 85% (switching). False positives vary between 12% (switching) and 17% (transmission).

In order to measure the quality of the information offered by the rooting module, two thresholds where established: one for the probability of nodes, and other for the probability of edges. With thresholds of 40% the percentage of false positives was 43% and true positives 86%; increasing the thresholds to 75% false positives reduced to 12%, and true positives decreased to 34%. Reducing the number of false positives is preferable since the main objective is offering useful information to the technicians. With this in mind, other type of metrics related to mean time to repair (MTR) can give a better understanding of the real benefits. MTR of tickets with a root cause was reduced about 27%; for certain types of

elements the reduction reached 41%. The number of tickets with a root cause that took longer than 24 hours to get fixed was reduced around 62%. For parent tickets, the reduction was 18%. This difference is due to the fact that most child tickets can be closed almost immediately once the parent-child relation is detected, whereas parent tickets need to be solved. Other significant advantages reported by technicians were: the reduction in the effort that they had to dedicate to searching for information, as the rooting module already showed all possible relations between tickets; valuable information concerning the number of elements affected by a root cause, and consequently, the possibility to know in advance the possible services affected by a failure, which in turn could be used to assign a more accurate severity level or priority.

Although in this initial phase the rooting module was not fully integrated with the TT system and did not perform any automatic action or offer recommendations, the results show that it provided technicians with valuable information to help them solve the tickets: all elements that could be affected by a given failure, and therefore, an indication of the severity of the problem; and, secondly, the possible existence of a root cause. To measure the technicians' level of satisfaction, they were asked to assign an evaluation mark about the utility of the information that the rooting module added to each ticket. The average mark was 6.8/10.

Figure 5 shows a real example graph with the information that the rooting module would attach to a ticket assigned to a switching element called "GR. RE CD34". The graph is centered on this element, the nodes above it represent the possible root causes, and the elements below it represent possible affected elements. Evidence about existing failures and existing parent-child relationships are shown with probability 100%.

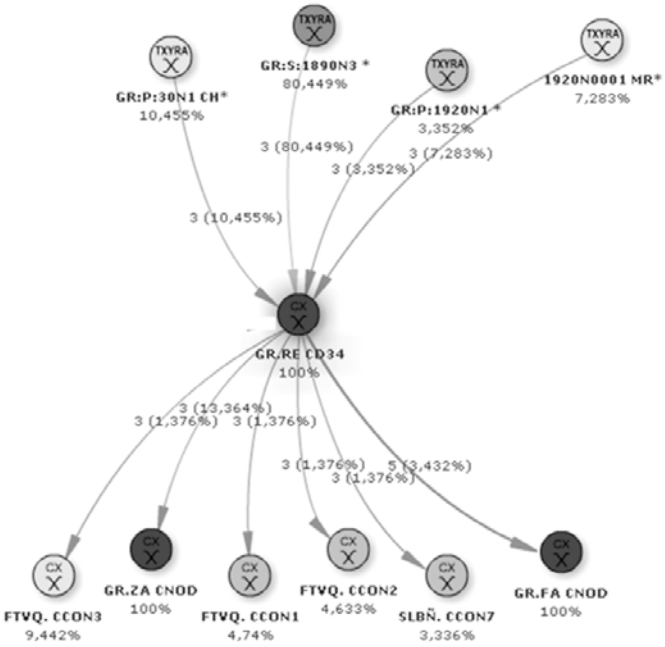

Figure 5. Information offered about element "GR. RE CD34"

A consideration that must be taken into account with this approach and should be addressed in future work is that the system has no information about an element until it fails for the first time, and even then the CPTs cannot offer sufficient confidence until sufficient statistical information is gathered.

 ISSN 1927-6974 E-ISSN 1927-6982

6 Conclusion

When systems are complex and comprise thousands of different interrelated elements of very different types, it requires an enormous effort to discover hidden associations. It requires expert technicians to execute many tests and search for probable root causes. This is a time consuming task that depends on the knowledge and experience of technicians.

An automatic rooting system attached to the main system can automatically collect all information compiled through years of operation and learn all relations between elements, which in turn can provide real time information that guides technicians in their daily work in different ways. It can warn of hidden root causes of failures, minimizing the time that services are inoperative, increasing user satisfaction, and decreasing maintenance costs. It can also inform of the scope of an existing failure, of its effect on related elements, and alert of a probable cascading of failures, preventing massive failure situations and allowing for an adequate assessment of the situation.

In all cases the rooting system is able to offer a graphical and numerical explanation, showing all elements involved in an incident, and confidence levels on the predicted information.

As a result, mean time to recovery (MTTR) can be reduced significantly, as well as the impact of a failure on the services offered to customers.

References

[1] http://www.bill-wilson.net/root-cause-analysis

[2] Jean, David R. (Palo Alto, CA, US), Marcopulos, Judy M. (Duluth, GA, US), Scherer, Rita H. (Birmingham, AL, US), 2007. Processes and systems for creating and for managing trouble tickets and work orders [Internet]. United States, AT&T Intellectual Property, Inc. (Wilmington, DE, US) 7289605. Available from: http://www.freepatentsonline.com/7289605.html

[3] Lewis, L., “Extending trouble ticket systems to fault diagnostics”, Network, IEEE, Nov. 1993.

[4] Lewis, L., “A case-based reasoning approach to the management of faults in communication networks”, INFOCOM '93. Proceedings. Twelfth Annual Joint Conference of the IEEE Computer and Communications Societies. Networking: Foundation for the Future, IEEE.

[5] Kiansing Ng and Huan Liu. “Customer Retention via Data Mining”. Artificial Intelligence Revision 14, 6 (December 2000), 569-590. http://dx.doi.org/10.1023/A:1006676015154

[6] Sasisekharan, R. et al, “Data Mining and Forecasting in Large-Scale Telecommunication Networks”, IEEE Educational Activities Department, New Jersey, USA. 1996: 37-43.

[7] Yaiza Temprado, Francisco Javier Molinero, Carolina García, Julia Gómez, "Knowledge Discovery from Trouble Ticketing Reports in a Large Telecommunication Company", 2008 International Conferences on Computational Intelligence for Modelling, Control and Automation; Intelligent Agents, Web Technologies and Internet Commerce; and Innovation in Software Engineering, 2008 (CIMCA); 37-42.

[8] Stefan Wallin, Viktor Leijon, Leif Landén, “Statistical analysis and prioritisation of alarms in mobile networks”, Int. J. Business Intelligence and Data Mining. 2009; 4(1).

[9] Julia Gómez , Yaiza Temprado , Margarita Gallardo , Carolina García , Francisco Javier Molinero, “Application of neural network to predict adverse situations in trouble ticketing reports”, IADIS Multi Conference on Computer Science and Information Systems. 2009.

[10] Kevin B. Korb, Ann E. Nicholson, CHAPMAN & HALL/CRC. “Bayesian Artificial Intelligence”.

[11] Daphne Koller and Nir Friedman, “Probabilistic Graphical Models: Principles and Techniques (Adaptive Computation and Machine Learning series)”.

[12] Dey, S., & Stori, J. “A Bayesian network approach to root cause diagnosis of process variations”. International Journal of Machine Tools and Manufacture. 2005; 45(1): 75-91. http://dx.doi.org/10.1016/j.ijmachtools.2004.06.018

[13] G. Weidl, A. L. Madsen, S. Israelson, “Object-Oriented Bayesian Networks for Condition Monitoring, Root Cause Analysis and Decision Support on Operation of Complex Continuous Processes: Methodology & Applications”.