A CASE-BASED REASONING METHOD FOR ALARM FILTERING AND CORRELATION IN TELECOMMUNICATION NETWORKS

Nahid Amani  
Iran telecommunication Research Center, End of North Kargar, Tehran, Iran  
email: n_ami@itrc.ac.ir

Mahmood Fathi  
Computer Eng. Department, Iran University of Science and Tech., Tehran, Iran  
email: mahfathy@iust.ac.ir

Mehdi Dehghan  
Computer Eng. Department, Amirkabir University of Tech., Tehran, Iran  
email: dehghan@ce.aut.ac.ir

Abstract

Alarm Correlation will play an important role in improving the service and reliability in modern telecommunication networks. As the network grows in size and complexity, the supervisors of network are finding it increasingly difficult to cope with the volume of alarm messages produced even from a single network fault. In this paper, a new using Case-Based Reasoning (CBR) method for alarm correlation in telecommunication networks has been developed. The proposed method has been simulated by developing three main modules: a module for generating faults and alarms, defining network configuration, and Alarm filtering and correlation by using Case-Based Reasoning. One of the most important aspects of the obtained results was the speed of the system. Because of its simplicity, the Case-Based Reasoning model is fast, requiring only a few floating-point calculations to produce the result. The accuracy of alarm correlation achieved in the simulation was higher than 90% in the case of unavailability of the required cases.

Keywords: Alarm Correlation, Alarm Filtering, Cased-Based Reasoning, Root Cause Analysis, Fault Management.

1 Introduction

In the last few decades, there has been increased demand for telecommunications services with the result that networks have grown in their size, complexity and bandwidth. Networks often consist of hundreds or even thousands of interconnected nodes from different manufacturers with various transport mediums. As a result, when a network problem or failure occurs, it is possible that a very large volume of alarm messages is generated. Therefore processing of the alarm for determines root cause of failure in network is very difficult.

Alarm correlation is fast becoming one of the primary techniques used to manage large volumes of event messages. It can reduce the amount of information presented to network operator by filtering transient or redundant alarms. So semantic content of the information presented can be increased by the correlation process, and root cause of failure will be detected faster and more exact. In this paper, we give a definition for alarm correlation and describe some of the requirements of an alarm correlation system, and using Case-Based technique for Alarm Correlation.

2 Definition Alarm Correlation

A telecommunication network consists of a number of interconnected components such as: switches, transmission equipment, etc. Each component in its turn contains several subcomponents. Each (sub) component and software module can produce alarms. They are messages describing some sort of abnormal situation; they do not necessarily indicate that is visible to the users. The number of produced alarms varies greatly, and they are in various types. Processing of the alarm flow is a difficult task for the following reasons [1]:

• The size of the networks and the diversity of alarm types that there are a lot of different situations that can occur.
• There is only little time for operators to decide what to do with each alarm.
• The hardware and software used in telecommunication networks develop fast. As new nodes are added to the network or old, ones are updated, the characteristics of the alarm sequences are changed. Thus the operators do not have time to learn what the appropriate response to each situation.
• A device may generate alarms due to a single fault.
• A single fault may be detected by multiple network components, each one of them emitting an alarm notification.
• The fault of a given component may affect several other components, causing the fault’s propagation. Alarm filtering and correlation can be solving the difficult of alarm processing with conceptual interpretation of alarm reports.

The term ‘correlation’ describes a situation in which two or more entities have a reciprocal relation [2-4]. The result of correlation process is two part: firstly, the total semantic content of the information is increased; secondly, the total number of ‘single units’ is reduced. Therefore, we can define alarm correlation as the interpretation of a multiple alarms so as to increase the semantic information content associated with a reduced set of messages. In the network management domain, alarm correlation is often used to help in the real-time
diagnosis of faults and fault localization. In a telecommunications network, we can correlate alarm messages which are reported even from a single network and reduce the amount of information reaching the operations staff.

2.1 Correlation Types

Several types of correlation may be identified, according to the operations executed on the alarms. The most important of these operations are detailed as follows [5, 6]:

2.1.1 Compression. In Compression phase, alarms are received in a time window and multiple occurrences of the same event, substituting the corresponding alarms for a single alarm.

\[ \text{Compression}[A, A, \ldots, A] = A \]  

(1)

2.1.2 Suppression. Suppression is a temporary inhibition of alarms referring to a given event, according to criteria (continuously evaluated by the correlation system) related to the dynamic context of the network management process.

\[ \text{Suppression}[A, B, p(A) < P(B)] = \phi \]  

(2)

2.1.3 Count. The substitution of a specified number of occurrences of an alarm, with a new alarm.

\[ \text{Count}[n \ast A] = B \]  

(3)

2.1.4 Generalization. Generalization consists of replacing an alarm, by the alarm corresponding to its super-class.

\[ \text{Generalization}[A, A \subset B] \Rightarrow B \]  

(4)

2.1.5 Specialization. Specialization is an operation which is the reverse of generalization, and that consist of substituting an alarm for another, corresponding to a sub-class.

\[ \text{Specialisation}[A, A \supset B] \Rightarrow B \]  

(5)

3 Case-Based Reasoning for Alarm Correlation

Case based reasoning is a method for problem solving that differs from other artificial intelligence (AI) methods since it bases its decisions on past experiences and situations (cases)[7,8]. In a case-based problem solver, new problems are solved by retrieving stored information about previous problem-solving episodes and adapting it to suggest solutions to the new problems. An important feature of case-based reasoning is capability for learning. When a problem is successfully solved, the parts of the solution which are likely to be useful in the future are stored. When an attempt to solve a problem fails, then the reason for the failure is identified and ‘remembered’ in order to avoid a recurrence of such a mistake. A case generally consists of information about the situation, the solution, the results of using that solution and some attributes that may be used in the searching for similar attributes of other cases. A case-based reasoning system may be described by a cyclic system that consists of four processes:

3.1 Retrieval

Given a problem, the most similar cases in the Case-Based have to obtain. There are a number of algorithms in use such as: Nearest neighbor, Inductive, Knowledge-based indexing,…

3.2 Reuse Methods

The reuse of the retrieved case solution in the context of the new case focuses on two parts:

• The differences among the past and current cases.
• What part of a retrieved case should be transformed and transferred to the new case?

The possible two subtasks of reuse are Copy and Adapt [9].

Copy: in simple classification task the differences are abstracted away and the solution class of the retrieved case is transferred to the new case as its solution class. This is a trivial type of reuse.

Adapt: there are two main ways to adapt past cases: reuse the past case solution (transformational reuse) and reuse the past method that constructed the solution (derivational reuse). In transformational reuse, the past case solution is not directly a solution for the new problem but there exists some knowledge in the form of transformational operators such that when applied to the old solution they transform it into a solution for the new case. Derivational reuse looks at how the problem was solved in the retrieved case. The retrieved case holds information about the method used for solving the retrieved problem. Now derivational reuse then re-instantiates the retrieved method to the new case and replays the plan into the new context.

3.3 Revise Methods

This phase consists of evaluating the case solution generated by the method. If successful, learn from the success; otherwise repair the case solution using domain knowledge or user knowledge. In evaluation phase, solution is applied in a real environment. Case repair involves detecting the errors of the current solution and retrieving or generating explanation for them.

3.4 Retain Methods

Save the new solution as a new case when a problem is solved by using a previous case.

3.5 When to Apply CBR?

• When a domain model is difficult or impossible to elicit
• When the system will require constant maintenance.
• When records of previously successful solutions exist.
• Or when similar problems show up often.

Fault Management is a set of functions which enables the detection, isolation and correction of abnormal operation of the telecommunication network and its environment [10, 11]. Fault Management is part of Maintenance Management and alarm
correlation is part of fault management. The most of failures that occur in telecommunication network usually are Repetitive. Therefore, we can use CBR for alarm Correlation.

3.6 Benefits of CBR

- It can solve problems within partially understood domains.
- Can handle new and changing data through their ability to use analogy.
- Can learn from experience through the acquisition of new cases.
- Do not require extensive maintenance.
- It can reason by analogy efficiently.
- Can use a method of knowledge acquisition that is less time consuming than expert system rule development.
- It is easier to build a case library than a Rule Based/ Bayesian Networks.
- Its knowledge representation is less restrictive.

The drawbacks of a CBR system are that it is unable to provide a solution to the given problem if there are no matches for the current situation in its case library. CBR systems unable to provide deep-level explanations for the proposed solutions. When the cases are unavailable, it will be time-consuming to build a case library.

4 Alarm Correlation Architecture

Alarm correlation using Case-Based Reasoning has been simulated that includes three simulators:
- Module for generating faults and alarms
  We have worked in real environment and our experiment is shown that creating alarms and faults is like to exponential distributed function therefore, this simulator using exponential distributed function for alarms and faults generation.
  \[ u = \text{randomuniform}(1) \]
  \[ x = -\frac{1}{\lambda} \log(u) \]
  \[ \text{return}(x) \]
- Module for defining network configuration
- Module for Alarm Filtering and Correlation by using Case-Based Reasoning

When a link fails, up to 100 and more alarms are generated and passed to the Operation and Maintenance Centre (OMC). An example is several alarms due to a link failure. The alarms are transmitted to the OMC. We define the produced alarm pattern as alarm vector, see “Fig.1”.

The simulator monitors alarms and faults and sort out the event from the symptoms. In other words, to find the fault from the generated alarms, see “Fig.2”.

Compression of Correlation simulator receive alarm sets in waiting time window that is computed with using of “(6),”

*Fig. 1. Alarms caused by a link failure*

\[ T_{s,t} = t_i + (1-\alpha)f_{t_{i+1}} + ... + (1-\alpha)^r f_{t_{i+r}} + (1-\alpha)^{r+1} f_{t_{i+r+1}} \]

*Fig. 2. Alarm filtering and correlation using Case-Based Reasoning*

In our experiences, we have applied the following similarity assessment scenario “(7),” in CBR for retrieve cases:

\[ \text{SIM}(\text{card}, \text{card}) = \frac{a \cdot \text{card}(E) + b \cdot \text{card}(D) + c \cdot \text{card}(U) + d \cdot \text{card}(U2)}{a \cdot \text{card}(E) + b \cdot \text{card}(D) + c \cdot \text{card}(U) + d \cdot \text{card}(U2)} \]

- E set of symptoms with the same values for Case1 and Case2
- D set of symptoms with different values for Case1 and Case2
- U1 set of symptoms with known values for Case1 but not for Case2
- U2 set of symptoms with known values for Case2 but not for Case1
- \( a=1, b=2, c=1/2, d=1/2 \) (the default weights used)
- Card cardinality of sets

All of process for detects root cause of the failure is following steps:

1. Input \( A[0], A[1], ... \) A[i]: vector of similar alarms \( i=0, ..., n \)
2. Compression \( A[0], A[1], ... \)
3. S = Suppression (A);
4. F = Count (S);
5. M = CBR_Retrieve (F);
6. N = CBR_Reuse (M);
7. J = CBR_Revise (N);
8. CBR_Retain (J);
5 Simulation Results

5.1 Evaluation of Compression Algorithm

The best choice of waiting time window is very important and depends on the Suppression parameters. These are deciding parameters that eliminate transient alarms. If the Suppression parameters are not selected appropriately and the length of waiting time window is selected too small, then most of the reported alarms will be detected as transient. In the case that length of waiting time window is selected too large, transient alarms may happen many times. So transient alarms are not detected well and error rate will increase, see “Fig.3”.

5.2 Evaluation of Root Cause of Failure Detection Algorithm

This algorithm works with three parameters:
- Threshold for acceptance of detected root cause of failure
- The weight of similarity assessment parameters
- The length of waiting time window

If first and second parameters are selected appropriately, the result of root cause of failure detection algorithm will be affected from changes of waiting time window length.

- If the length of window is selected too small then there is not enough time for receiving alarms set and similarity between alarms set and cases in CBR will decrease so suite case will not be retrieved. Therefore root cause of failure detection will not be done correctly and error rate will increase, see “Fig.4,” and “Fig.5”.
- If length of window is selected too large, then many faults may happen in the network and the volume of produced alarm messages will increase, so similarity between alarms set and cases in CBR will decrease therefore suite case will not be retrieved and error rate will increase, see “Fig.4,” and “Fig.5”.

If threshold is selected too small then even with lowest similarity value, each case is retrieved from case based therefore system overhead and error rate will increase, see “Fig.6,” and “Fig.7”.

If threshold is selected too large, then it is possible that all alarms set will not receive in that time window, so similarity between alarms set and cases in CBR is less than threshold value.

Therefore suite case will not be retrieved and error rate will increase, see “Fig.6,” and “Fig.7”.

Fig. 3. Relation between waiting time window and error percent

Fig. 4. Relation between waiting time window and fault detection percent

Fig. 5. Relation between waiting time window and error percent
In this paper, we described an algorithm for alarm filtering and correlation in telecommunications networks that uses Case-Based Reasoning technique. The proposed method has been simulated by developing three main modules: a module for generating faults and alarms, a module for defining network configuration, and a module for Alarm filtering and correlation by using Case-Based Reasoning. One of the most important aspects of the results was the speed at which they were obtained. Because of its simplicity, the Case-Based Reasoning model is fast, requiring only a few floating-point calculations to produce a result. Simulator for Alarm Correlation performs with 100% precision when no input errors are present and performance. Thereafter, the performance drops slightly to 90% precision when the all cases are unavailable in case-base.

6 CONCLUSION

References


