An Intelligent Alarm Management System for Large-Scale Telecommunication Companies

Raúl Costa[1,2], Nuno Cachulo[2], Paulo Cortez[1]

[1] Department of Information Systems/Algoritmi R&D Centre, University of Minho, 4800-058 Guimarães, Portugal
[2] PT Inovação, Rua Eng. José Ferreira Pinto Basto, 3810-106 Aveiro, Portugal
{raul-s-costa, est-n-cachulo}@ptinovacao.pt, pcortez@dsi.uminho.pt

Abstract. This paper introduces an intelligent system that performs alarm correlation and root cause analysis. The system is designed to operate in large-scale heterogeneous networks from telecommunications operators. The proposed architecture includes a rules management module that is based in data mining (to generate the rules) and reinforcement learning (to improve rule selection) algorithms. In this work, we focus on the design and development of the rule generation part and test it using a large real-world dataset containing alarms from a Portuguese telecommunications company. The correlation engine achieved promising results, measured by a compression rate of 70% and assessed in real-time by experienced network administrator staff.

Keywords: Network Management, Association Rules, Event Correlation.

1 Introduction

During the last decades, with the growth of the Internet and cellular phones, there has been an increase in telecommunications demand. To support this growth, telecommunication companies are investing in new technologies to improve their services. In particular, real-time monitoring of infrastructures and services is a key issue within any telecommunication operator. On one hand, the quality of service has to be assured by a timely fault diagnosis with evaluation of service impact and recovery. This is particularly needed to fulfill the Service Level Agreements (SLAs) that are set between the service provider and customers. On the other hand, operational tasks must be simplified to guarantee reduced OPerating EXpenses (OPEX). Hence, there are a wide variety of software tools and applications that were developed to address this issue [5][19] and most of these tools require a human intervention for corrective action. In particular, several of these solutions incorporate correlation engines, which is the "smart" part of the management platform. The goal is to exploit data collected by monitoring subsystems, as well as notifications sent spontaneously by managed entities. However, as networks become larger, with more intricate dependencies and dynamics, new challenges are posed to these correlation engines, such as [11]:

(i) Large telecommunication companies' networks are very heterogeneous and event correlation rules rely heavily on information provided by the vendor, which may not be always available. Hence, the effectiveness of these solutions depends heavily on the business relation with the device vendor.
(ii) Most paradigms for rule correlation, including rule-based reasoning, probabilistic reasoning, model-based reasoning and case-based, are configured out of the box (i.e. predefined) and they do not learn with experience, thus they cannot respond properly to new situations.
(iii) "Keeping users in the loop", i.e., maintaining users informed of changes in service availability is not always done properly. Therefore, it is difficult to assess effectiveness of the results on problem reporting. What if trouble ticketing does not point the correct root cause for a certain situation or what if the problem description is incomplete and misses important event details?

Ideally, an intelligent alarm management system should be capable of parsing the massive amount of received alarm events while reducing human intervention. Yet, designing and maintaining such an ideal system is not a trivial process. Traditional event correlation paradigms do not work well when the managed domains that they cover are large-scale and dynamic (i.e. change through time) [18].

Advances in information technologies have made it possible to collect, store and process massive, often highly complex datasets. All this data hold valuable information such as trends and patterns, which can be used to improve decision making and optimize chances of success. Data mining techniques, such as association rules [4], aim at extracting high-level knowledge from raw data [22]. The goal of this paper is to study the feasibility of building a decision support tool for large telecommunications operators using data mining algorithms. Similar to market-basket data sequences [7], we will assume that a network generates sets of events that are related to the same situation. For example, in the access network of GSM systems, the Base Station Subsystem (BSS) contains Base Stations (BS) that are connected via a multiplexing transmission system to the Base Station Controller (BSC). These connections are very often realized with microwave line-of-sight radio transmission equipment. Heavy rain or snow can temporarily disturb the connections between the antennas. The temporary loss of sight of a microwave disconnects all chained BS from the BSC and results in an alarm burst. Our goal is to automatically discover these patterns. When this information is combined with other features (e.g. trouble-ticket data) it is possible to generate rules that lead to alarm correlation and root cause analysis. The data mining algorithms can generate these rules automatically, from the collected data, in opposition to traditional systems where the rules are pre-coded or manually configured. Thus, our approach may have a large impact in the business, since it can potentially save the company a considerable amount of lost revenue. A timely and correctly identification of the root cause will aid in the anomaly correction, reducing the maintenance costs. Furthermore, by providing a better service, it is expected that the costumer complaints will be reduced, diminishing the costumer risk of leaving the operator.

In this paper, we propose an architecture that is targeted to address an adaptive and self-maintained alarm correlation system. In particular, we are working on an automatic rule discovery approach applied to a large telecommunications operator. Our approach adapts an association rule algorithm (i.e. Generalized Sequential

Patterns) to the telecommunication domain. It has the advantage of being independent of the network topology and uses trouble-ticket information to get feedback from the event correlation results. The final aim of this line of research is to incorporate a reinforcement learning system that will guide the association rule generation and selection. As such, this is a milestone where implementation results are assessed and overall architecture presented for the first time.

2 Related Work

The concept of sequential pattern was introduced by Agrawal and Srikant [3] from a set of market-basket data sequences, where each sequence element is a set of items purchased in the same transaction. The same researchers also proposed in [2] the Generalized Sequential Patterns (GSP) algorithm, which allows a time-gape constraint, where an item from a given sequence can span a set of transactions within a user-specified window. In addition, the algorithm allows that the item can cover different taxonomies.

Mannila et al. [16][17] presented the WINEPI framework for discovering frequent episodes from alarm databases, allowing the discrimination of serial and parallel episodes. Later, this framework was adopted to analyze Synchronous Digital Hierarchy (SDH) [9] and GSM [21] network alarms. In [24][26], Wang et al. studied the data mining of asynchronous periodic patterns in time series data with noise, proposing a flexible model, based on a two-phase algorithm, for dealing with asynchronous periodic patterns. They only considered the model in the time series domain and did not consider other periodic patterns in the data.

Sequential algorithms are helpful for mining alarm databases in order to support the creation of rule-based expert systems. Rules like "'If A and B then C'' are the main approach for solving the alarm correlation problem and root cause analysis. An advanced event correlation system is the EMC Smarts Network Protocol Manager [8], which uses the patented "Code Book". It works as a black box, retrieving a problem from a set of symptoms. The drawback of this solution is that the rules are pre-coded (i.e. static) and rely heavily on information provided by the vendor.

A problem that often comes from the usage of unsupervised learning processes, such as the popular Apriori algorithm, is the large quantity of rules generated. In order to select only the most interesting rules it is necessary to select proper criteria to filter and order them. Several measures of interestingness have been proposed within the context of association rules (e.g. confidence, support and lift). Choosing an adequate metric is a key issue for the success of generating useful rules.

There are also other techniques that have been applied by several commercial network management solutions. Smart-Plugins (SPIs) are being utilized for managing new devices and protocols. These plugins extend the base management capabilities with specific vendor/technology functionalities. HP OpenView's Network Node Manager (NNM) [13] offers a broad multi-vendor device coverage by allowing deployment of several SPIs. However, often the required information to develop SPIs is difficult to get due to the existence of proprietary protocols and technology.

In an attempt to resume the state-of-the-art in network management and present future perspectives, Gupta [11] listed several innovative frameworks, such as EMC SMARTS and SPIs that are being used for advanced root cause analysis. As a future direction, Gupta suggests the implementation of environment aware network management solutions. The concept is to use historical data collected from the network to help identify anomalous situations from the deviation of the traditional patterns. Expert systems are also pointed as a way to reduce human intervention. Such a system would learn from human experience and assist the proposal for repair actions. It could also be used to teach standard procedures to new team members.

Martin et al. [18] explain why event correlation is still an open issue. The complexity of event correlation has increased over the last few years. Current algorithms make inappropriate simplifying assumptions and new models, algorithms and systems are required to deal with such complex and dynamic networks. To overcome current limitations, it is necessary to improve network information in order to build learning models to assist infrastructure managers. Such models are required to deal with uncertain knowledge and learn from past experience.

Often, equipment vendors are responsible for specifying alarm's parameters, and telecommunication companies do not have the flexibility to adapt them to their specific necessities. For example, alarm severity is defined in the X.733 standard (ITU-T, 1992). It would be expected that severity level is closely related to the malfunction priority, but this is not always true. Assigning a malfunction's priority relies heavily on network's administrator's experience, depending on a dynamic evaluation of redundancy, network topology and eventual SLAs. Wallin and Landén [23] proposed a solution that uses neural networks to automatically assign alarm priorities. The authors point that the advantages of using neural networks come from their good noise tolerance capabilities and the ability to learn from network administrator staff by using the manually assigned priorities in trouble ticket reports.

3 Proposed Architecture

A common fragility of most paradigms for event correlation is the lack of ability to learn with experience or adapt to situations that have not been pre-coded. In contrast, we propose an alarm management system that automatically extracts correlation rules from historical alarm data. Changes in the network will not affect the correlation performance, since the extraction process will run periodically. The information is provided to network administrators for supporting the associated trouble tickets management. By supervising the trouble ticket lifecycle we gain feedback on the rules' performance and this information can be used to adjust selection from the pool of available rules.

Within the considered telecommunications operator (PT Inovação), the Fault Management and Fault Reporting modules work independently. Some interface features are used to enhance trouble ticket generation, but users still have to manually select related alarms and indicate one that is pointed to be the cause the problem. This type of system is quite similar to what can be found in the majority of the network operating centers.

The following steps can describe the telecommunication alarm management tasks:

- parse the received alarms and group the ones related to the same problem;
- associate alarms with a trouble ticket to manage the problem resolution process and indicate the cause (i.e. to identify the root cause);
- assign a priority to the trouble ticket; and
- analyze and fix the problem.

To achieve the above goals, we proposed an integrated intelligent management architecture that is illustrated in Fig. 1. The main components of the system are:

- **Fault Management Platform** – collects, processes and displays alarm events via a Web GUI integrated in the portal Server.
- **Fault Reporting Platform** – records and forwards fault reports, also known as TTKs (Trouble Tickets), to suppliers.
- **Preprocessing** – is responsible for the correct linking of all the values in order to create valid records. It proceeds with the copy of the values entered by network administrators during fault reporting activity and also alarm events.
- **Assurance Warehouse** – stores information about alarms (events plus user operation logs) and TTKs creation and management, with information on intervention or unavailability.
- **Rules Generator** – generates rules for alarm correlation and root cause analysis from off-line training using the data from the warehouse. Rules can also be created by a human editor. It runs with a predefined periodicity (a configuration parameter) in order to allow the replacement of poor performing rules. Possible algorithms are Case-Based Reasoning [9], a solving paradigm that relies on previously experienced cases, and association rules (addressed in this paper).
- **CORC Rule Database** – stores the correlation and root cause analysis rules along with several evaluation metrics. The evaluation metrics include support, lift, support and information on reception order. These statistics are updated every time new information is collected.
- **Reinforcement Learning Module** – refines the rules database based on feedback obtained from the Portal Server. This process receives feedback from two distinct ways: after the resolution of the network malfunction reported in TTK, it is possible to evaluate if proposed correlation and root cause were correct, by means of text mining [6]; also, it is expected that all alarms related to one malfunction cease nearly at the same time and immediately after its root cause resolution.

- **Alarm Prioritization** – it uses a supervised learning algorithm (e.g. neural networks) to learn from alarm priorities assigned by network administrators in TTK. This module has been already been proposed in [23] with a 71% success rate. This information is passed to users and is also used so that in overflow situations the most important alarms are processed first by the management mechanism.
- **Management** – receives real-time alarms from the fault management and processes them. A business rules engine is included to support output from the Rules Management System and from Alarm Prioritization and returns decision-making logic. The enriched result is then passed to the Fault Management platform that has a dedicated web-based GUI to present correlated alarms with visual indication of root cause and assigned priority.

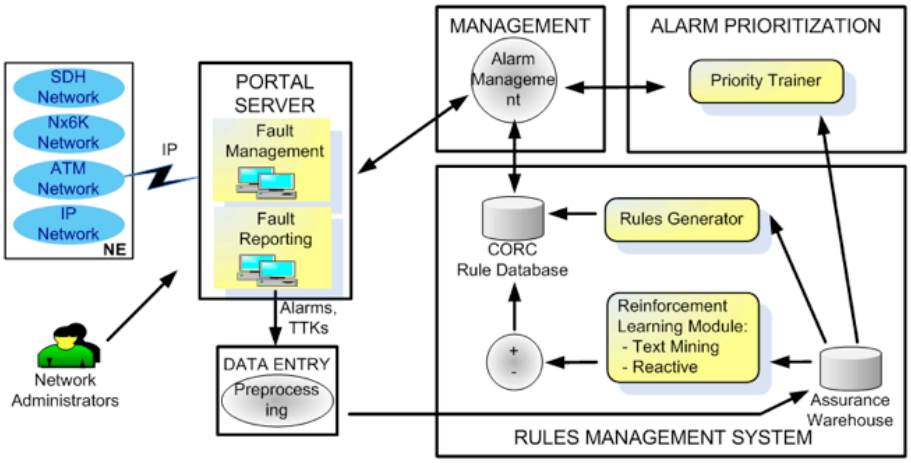

Fig. 1. The proposed intelligent alarm management architecture.

As indicated in the figure, we propose feedback loops for making correlation and root cause analysis. The first loop is periodically executed and involves calculating candidate rules by several specialized association rules algorithms utilizing more static information on the alarm history. The output of the algorithms is combined in one rule database which is used to dynamically associate related alarms and select root cause. In the second loop we continuously gather and evaluate user reaction to the presented suggestions. The learning module uses this information to refine the associations and root causes in the database and thus to immediately affect the selection of future results. In the present paper, we will focus only in the automatic correlation rule discovery part of the proposed architecture (module Rules Generator from Figure 1).

4 Materials and Methods

4.1 Alarm Data

This study will consider an alarm database from a major Portuguese telecommunication company. The data was collected from PT Inovação's alarm management system, called Alarm Manager. This platform's architecture follows the most recent guidelines for the implementation of next generation Operations Support Systems (OSS), as laid down in TM Forum's NGOSS [20]. It was developed in Java and contains J2EE middleware. This system is operating stably for more than 2 years, dealing daily with huge amounts of data collected from heterogeneous networks including access, aggregation/metro and core segments of transport and technologies, such as IP/MPLS, CET, ATM, SDH, PDH, DSL or GPON.

The dataset contains the history of alarm events that occurred from March 2007 to November 2008. Each alarm record already includes a basic type of correlation that aggregates all events if they represent repetitions of an active alarm with the same description and entity. The beginning and termination of the alarm occurrence, along with other events that occurred in between, are registered as a single record.
The overall table contains near 15 million of rows. In addition to identification of related fields, it contains several attributes that describe an alarm event. Thereby, it has several time occurrence values, including the starting and ending time and date of an alarm. Other time stamps are the moment where the alarm changed its state, the time it was recognized by a network supervisor and the time it was archived in the history table. Note that only the alarms terminated have this last time stamp and these are the ones that will be considered in this work. Moreover, the database contains other attributes, such as: ending type (an alarm can be terminated normally, manually by a network administrator or automatically by configuration on the management system); the number of events that each alarm aggregates; network domain and, finally, a description of the specific problem that the alarm reports.

Before applying the data mining algorithms, we first performed a cleaning and data selection stage. Experts from PT Inovação were consulted in order to discard irrelevant variables. Some statistical tests were additionally performed (e.g. Pearson correlation [1]) to check each attribute independency. Furthermore, we noticed that certain attributes had a very limited range of values and were not strictly related to the alarm occurrence, thus these were also discarded. Finally, we discard all topological attributes, as our approach does not require such data and we do not want to constrain the data mining process. Table 1 resumes the selected attributes that will be used in the analysis.

We performed additional preprocessing operations, after a feedback that was obtained from the network management people. For instance, we removed alarms terminated manually by network managers or automatically by system configuration, since these suffered from an abnormal ending. In addition, the **Specific Problem** records were reported in two different languages (i.e. English or Portuguese). Thus, we standardized these records by using auxiliary information that was available in supplied catalogs.

Table 1. Summary of the selected alarm data attributes.

Attribute	Description
Alarm ID	Alarm identification, an information that can be used to obtain the order of the alarms arrival into the Alarm Manager.
Event Counter	The number of alarm events aggregated into a single alarm.
Starting Time	The initial time of the alarm.
Ending Time	The ending time of the alarm.
Specific Problem	The specific problem occurred in the network that the alarm carries with it.
End Type	Information on how the alarm was terminated.

4.2 Methods

We will base the alarm correlation and root cause detection on the events' time and its characteristics. This approach is closely related to the problem of finding sequential patterns in large amounts of data. Therefore, we adopted the GSP algorithm, which already has an implementation in the open source data mining tool Weka [25]. The GSP algorithm uses the concept of a sliding window to group raw data into candidate sequences of alarms that regular association algorithms, such as the well-known Apriori, are capable to use. Apriori based algorithms are then able to find the sequential patterns that we aim. However, the GSP was not specialized enough. More context orientation was required for pre-processing and candidate sequence extraction stages. Not to mention that the issue of root cause is obviously not covered by this algorithm and needs to be addressed here.

The first obstacle in applying our approach came from the high number of consecutive alarms, or very close in, having the same specific problem. This happens because a single problem might generate errors in several dependent entities. For example, the same Alarm Indication Signal is sent for all the ports in a SDH slot when the connected STM is down with a Loss of Signal. Thus, we decided to aggregate similar alarms to obtain rules containing only different ones, updating the event counter for the resulting alarm to be used in candidate extraction. The sliding window is used to modulate data and the time window for this aggregation should be equal to the maximum interval defined for timely windows. An example of the data to sequence extraction is represented in Tables 2 and 3.

At this point it is possible to extract candidate sequences. To achieve this, sliding windows were adopted for the starting time and ending times. This way, two or more alarms are only aggregated in the same sequence if both the starting and ending times fit the sliding window. This should highly improve the confidence of candidate extraction. Also, a degree of flexibility is considered so that sliding windows can adapt to particular situations. Based on preliminary experiments, we set heuristically a minimum interval of three seconds. This interval was corroborated by a network expert as a recommended value. Nevertheless, if a sequence is being constructed and this time is reached, the sliding window is extended to a maximum value in which the found alarms are joined into the sequence. Again, for the same reasons, this value was

set to five seconds. This process is done over the preprocessed data ordered by the alarms starting date. Result sequences, after applying the described methods to the previous example dataset, would be:

- Sequence 1: B / A / C;
- Sequence 2: D / E;

Before candidate sequences proceed to the rule finding phase, they are sorted to ensure that the order of constituting elements is always the same. For instance, we want that the sequence B / A / C is considered equally as the sequence A / B / C, because the relation between the constituting events is the same.

Table 2. Example of the original dataset.

Specific Problem	Starting Time	Ending Time	Event Counter	Alarm ID
B	00:00:01	00:00:06	1	2
A	00:00:01	00:00:06	1	1
B	00:00:01	00:00:06	1	4
B	00:00:01	00:00:06	1	3
C	00:00:03	00:00:07	1	5
D	00:00:10	00:00:15	2	6
E	00:00:13	00:00:15	2	7
B	00:00:01	00:00:06	1	2

Table 3. Example of the dataset result after event aggregation.

Specific Problem	Starting Time	Ending Time	Event Counter	Alarm ID
B	00:00:01	00:00:06	3	2
A	00:00:01	00:00:06	1	1
C	00:00:03	00:00:07	1	5
D	00:00:10	00:00:15	2	6
E	00:00:13	00:00:15	2	7

By now, the information on probable root cause for each built candidate sequence must be considered. In our analysis, root cause calculation is based in two factors: the first alarm of a sequence found in time and the number of events aggregated by each alarm. It is frequent to have several events with the same generation time stamp, because time precision is measured in milliseconds. For these situations, first event received in Alarm Collection Gateway is considered. This solution has some risks that might mislead root cause analysis due to different event propagation times in the network (the protocol most used is SNMP). However, considering the vast dataset used for modeling, there are some guaranties of attribute relevance. Moreover, in particular analysis scenario, there is an acknowledge (ACK) mechanism between monitoring agents and alarm collection which guaranties delivering and correction of order over SNMP's connectionless protocol. On other hand, the usage of event counters is based on the observation that typically the root cause is a single alarm

event that can cause a wide number of alarm events. These two aspects are combined in an editable rule system, which is used to classify the probability of the root cause for each sequence. For example, if we have a rule where the percentage of an alarm being the first to appear is very high, then it is highly probable that this is the root cause. Also, if the percentage of occurrences is low, when compared to the remaining events of the sequence, then this increases the probability of being the root cause. If none of these situations occur, the confidence of the suggestion is set to a negligible value. Going back to our example dataset, we had the following root cause information for our sequences:

- Sequence 1: First Alarm – A, Event Counter – A:1,B:3,C:2;
- Sequence 2: First Alarm – D, Event Counter – D:2,B:2;

After founding a frequent rule, all the information related to managing root cause selection must be updated. We have now candidate sequences and the last step is to call the Apriori algorithm to build and classify the resulting association rules. Apriori bases its behavior on the support of candidate sequences, by removing candidates that are below a given threshold. The use of Apriori instead of a simple counting of candidates is justified by its main principle that increases the rule finding scalability: if a sequence is frequent then all of its subsets should also be frequent. In other words, if a sequence is infrequent then all the sequences that contain that sequence can be pruned and not taken into process. The Apriori algorithm implemented also computed additional metrics, such as the confidence and lift, for a more detailed rule evaluation, as we are going to describe in next section.

4.3 Evaluation

As said before, the classic Apriori algorithm only classifies a sequence for its support, which is clearly insufficient for our purpose. We do not only want to know high frequent sequences present in data, but also less frequent ones that also apply as good modeling rules. In fact, these rules are also very interesting in terms of business value because as rare as rules become, the more difficult is for network administrators to discover their underlying behavior. The inheriting risk is a possible overflow of rules, so it is mandatory to define a minimum support threshold.

In order to classify the discovered rules, we will adopt two additional measures, well known within the association rules context, confidence and lift, and try to adequate their meaning to the present problem of discovering sequential alarm patterns. Lift provides information about the change in probability of the consequent in presence of the antecedent. The ‘IF’ component of an association rule is known as the antecedent. The THEN component is known as the consequent. The antecedent and the consequent are disjoint; they have no items in common. Both metrics depend on the division of the association rule in two parts (the antecedent and the consequent) but this is not really consistent with the rules we want to obtain, where there is only an unordered sequence of alarms that are somehow correlated. To make possible the use of this metric, we assume that the rule’s lift is the maximum possible for that sequence. In other words, we find the antecedent that maximizes the

confidence rule metric, because its formula depends on the antecedent support. This is done because it is more harmful to lose a good rule than to over classify one. We have to keep in mind that the rules will suffer posterior evaluation from feedback results and even might get eliminated by the reinforcement learning system (not presented in this work but to be addressed in the future). The confidence of a rule indicates the probability of both the antecedent and the consequent appear in the same transaction. Confidence is the conditional probability of the consequent given the antecedent. This gives us a better classification for the items within a rule than the simple support of the occurrence of all of them together. Confidence can be expressed in probability notation as the support of A and B together dividing by the support of A:

$$\begin{aligned} &\textbf{Confidence}\ (\text{A implies B}) = P\,(B/A), \text{ or} \\ &\textbf{Confidence}\ (\text{A implies B}) = P(A, B) / P(A)\ . \end{aligned} \qquad (\mathbf{1})$$

Regular confidence implementation cannot be used for Apriori because we are not interested in finding rules with antecedents and consequents. These are more appropriate, for instance, within the retail industry, where it is relevant knowing the difference between rules such as “buying milk implies buying cereal” or vice-versa. In our case, we calculate the highest confidence of a rule, selecting the possible antecedent for a rule with least support.

Both support and confidence will be used to determine if a rule is valid. However, there are times when both of these measures may be high, and yet still produce a rule that is not useful. If in our case the support of the rule consequent, not used in calculation of confidence, is very high then we are producing a rule that, even with high confidence, is not very interesting because it is normal that the antecedent is commonly seen with the consequent, given the fact that the consequent is very frequent. Thus, a third measure is needed to evaluate the quality of the rule. Lift indicates the strength of a rule over the random co-occurrence of the antecedent and the consequent, given their individual support. It provides information about the improvement, the increase in probability of the consequent given the antecedent. Lift is defined as [12]:

$$\textbf{Lift}\ (\text{A implies B}) = (\text{Rule Support}) / (\text{Support(A)} * \text{Support(B)}) \qquad (\mathbf{2})$$

Any rule with an improvement (lift) of less than 1 does not indicate an interesting rule, no matter how high its support and confidence are. Lift will be used as another parameter that classifies a rule and gives it more reliability. For example, a rule with low confidence but high lift has a higher degree of reliability than if we are just looking for the confidence metric. The computation of lift in our work is based in the same purpose used to calculate confidence, using the earlier defined antecedent and consequent of a rule to get the needed values for lift formula.

5 Current Results

To test our approach, we considered a recent sample of the alarm database, from 1st of March 2009 until the 15th of the same month, with a total of approximately 2.4 million of alarm events. The dataset was further divided into two subsets: the first 2/3

of the data was used to extract the candidate sequences, while the remaining 1/3 was used for testing. In the training stage, minimum thresholds considered were: for support 1%, confidence 40% and lift 1. The minimum sliding window time interval was restricted to 3 seconds. Under this setup, the rule that aggregated a maximum number of alarm types contained 7 elements and 268 rules were discovered. The rules with maximum support and lift are presented in Fig. 2. The first result shows a sequence of two alarms IMALINK (IMA Link Error) and LOC (Loss of Cell Delineation). Due to monitored network's characteristics, this sequence occurs very frequently, with a support of 1057 and a lift above 2. This sequence represents the situation when a group of virtual ports have an ATM layer above them working as a single communication channel. One problem in a single port is sufficient to affect all others in the same group and also the ATM layer. So, in some error situations, virtual ports send IMALNK alarms and the LOC is related to the ATM layer.

2 Alarm Types		SUPPORT	LIFT					
IMALNK	LOC	1057	2,02					
LP-PLM	LP-UNEQ	81	67,89					
3 Alarm Types			SUPPORT	LIFT				
SD	IMALNK	LOM	1190	2,01				
OOF	HP-RDI	MS-RDI	40	106,54				
4 Alarm Types				SUPPORT	LIFT			
OOM2	IMALNK	SD	LOM	617	1,74			
OOM2	TU-AIS	AU-AIS	DNU	7	108,37			
5 Alarm Types					SUPPORT	LIFT		
OOM2	IMALNK	SD	LOM	LOF	87	1,71		
HP-RDI	LOS	DNU	MS-RDI	TU-AIS	4	41,55		
6 Alarm Types						SUPPORT	LIFT	
OOM2	OOM1	IMALNK	SD	LOM	LOF	14	1,99	
OOM2	AIS	IMALNK	LP-PLM	LP-UNEQ	TU-AIS	4	2,54	
7 Alarm Types							SUPPORT	LIFT
OOM2	AIS	LP-RDI	IMALNK	LP-PLM	LP-UNEQ	TU-AIS	4	1,53
RAI	OOM2	LP-RDI	IMALNK	SD	LOM	LOF	3	4,63

Fig. 2. Examples of discovered association rules (with highest support and lift values).

A prototype was implemented using JRules, an open source and standards-based business rules engine [14]. The previous discovered rules were adapted to fit JRules requirements and then events from the testing set were injected into the rules engine, simulating a real-time alarm collection scenario. This was performed for a total of 501.218 events. The objective was to measure the degree of compression of the alarms that would be presented to end users, if using our correlation method, and to validate the real significance of discovered sequences.

In result of compression using the discovered correlation rules, the number of correlated alarms was 158.191, achieving a compression rate of near 70%. This rate is similar to the results achieved by the currently used systems [27]. In correlation by compression, alarm events from the same alarm type and regarding the same entity are grouped, as well as end notifications of respective alarms.

We asked a very skilled network administrator to validate the discovered rules with higher support and lift values (from Figure 2). In this expert's opinion, this approach is very interesting, "as it reveals certain patterns not strongly presented in

network protocols, but discovered with experience". Also, this solution enables a service oriented monitoring which is clearly more effective than the current single network entity approach. Regarding the previous described example (rule for IMALINK and LOC), during the performed runtime experiments our correlation engine grouped sequences of 4 to 8 alarms of type IMALNK associated with 1 LOC alarm. In this way, an immediate action can be set in order to create the respective trouble ticket, which launches the reporting of this malfunction to the operational staff. While reducing the supervisor's parsing work, the process of reparation is also speeded up, contributing to better service levels.

6 Conclusions

Fault management is a critical but difficult task in network management. The flow of alarms received by a management center should be correlated automatically to a more intelligible form, in order to facilitate identification and correction of faults. Unfortunately, the construction of an alarm correlation system requires high expertise and developing time. In opposition to current adopted solutions, with static pre-coded rules, we propose a smarter system that is able to learn from raw data and based in data mining techniques (for rule generation) and reinforcement learning (for rule selection). In particular, we focus on one component of the proposed architecture, the rule generator. A prototype was implemented to test it in a J2EE environment with real-world data from a large Portuguese telecommunications operator. A total of 15 millions of alarm records were used for training and testing, resulting in 268 association rules. The employment of correlation methods compressed the total amount of events with a 70% rate and the network administrator feedback was that there is a great potential to reduce operational costs, fault identification and reparation times. In future work, we intend to explore additional measures of interest for rule selection (e.g. Loevinger rank) [15]. We also expect to develop the remaining components, until the final architecture is implemented and tested in a real-world environment.

References

1. Aczel, A., *Statistics - Concepts and Applications*, IRVIN, 1996.
2. Agrawal, R. and Srikant, R., Fast Algorithms for Mining Association Rules, In *Proceedings of 2nd Int. Conference on Very Large Databases*, pages 487-499, 1994.
3. Agrawal, R. and Srikant, R., Mining Sequential Patterns, In *Proceedings of Eleventh International Conference on Data Engineering*, pages 3-14, 1995.
4. Agrawal, R. and Srikant, R., Mining Sequential Patterns: Generalizations and Performance Improvements, In *Proceedings of Advances in Database Technology - EDBT'96: 5th Int. Conference on Extending Database Technology*, Springer 1996.
5. Bernstein, L. and Yuhas, C.M., Basic Concepts for Managing Telecommunications Networks: Copper to Sand to Glass to Air, Academic/Plenum Publishers, 1999.
6. Berry, M.W., Survey of text mining: clustering, classification, and retrieval, Springer, 2003.
7. Brin, S., Motwani, R. and Silverstein, C., Beyond market basket: Generalizing

association rules to correlations, In *Proceedings of ACM SIGMOD International Conference Management of Data*, pages 265–276, 1997.
8. EMC Corporation, EMC Smarts Network Protocol Manager, available at: http://www.emc.com/products/detail/software/network-protocol-manager.htm, accessed April, 2009.
9. Gardner, R.D. and Harle, D.A., Methods and systems for alarm correlation, In *Proceedings of GLOBECOM'96*, volume 1, pages 136-140, 1996.
10. Gardner, R.D. and Harle, D.A., Fault Resolution and Alarm Correlation in High-Speed Networks using Database Mining Techniques. In *Proceedings of Int. Conference on Information, Communications and Signal Processing*, volume 3, 1997.
11. Gupta, A., Network Management: Current Trends and Future Perspectives. *Journal of Network and Systems Management*, 14:483-491, 2006.
12. Hahsler, M., Grün, B. and Hornik, K., arules – A Computational Environment for Mining Association Rules and Frequent Item Sets, *Journal of Statistical Software*, 14(15):1-25, 2005.
13. HP Corporate, HP Network Node Manager (NNM) Advanced Edition software, available at: http://www.openview.hp.com/products/nnm, accessed April, 2009.
14. JBoss devision of Red Hat, JBoss Rules, available at: http://www.jboss.com/products/rules/, accessed April, 2009.
15. Lenca, P., Meyer, P., Vaillant, B. and Lallich, S., On selecting interestingness measures for association rules: User oriented description and multiple criteria decision aid, *European Journal of Operational Research*, 184(2): 610–626, 2008.
16. Mannila, H. Toivonen, H. and Verkamo, A., Discovering frequent episodes in sequences, In *Proceedings of 1st Conference on Knowledge Discovery and Data Mining (KDD-95)*, pages 210-215, 1995.
17. Mannila, H. and Toivonen, H., Discovering Generalized Episodes Using Minimal Occurrences, In *Proceedings of Second International Conference on Knowledge Discovery and Data Mining*, pages 146-151, 1996.
18. Martin, J.P., Flatin, M., Jakobson, G. and Lewis, L., Event Correlation in Integrated Management: Lessons Learned and Outlook, *Journal of Network and Systems Management*, 15(4):481-502, 2007.
19. Raman, L., OSI Systems and Network management, *IEEE Communications Magazine*, 36(3):46 – 53, 1998.
20. TM Forum Association, TM Forum Solution Frameworks (NGOSS) Overview, available at: http://www.tmforum.org/Overview/1912/home.html, accessed April, 2009.
21. Tuchs, K. and Jobman, K., Intelligent Search for Correlated Alarm Events in Database, In *Proceedings of IFIP IEEE International Symposium on Integrated Network Management*, pages 405-419, 2001.
22. Turban, E., Sharda, R., Aronson, J. and King, D., *Business Intelligence: A Managerial Approach*, Prentice-Hall, 2007.
23. Wallin, S. and Landén, L., Telecom Alarm Prioritization using Neural Networks, In *Proceedings of 22nd International Conference on Advanced Information Networking and Applications,* pages 1468-1473, 2008.
24. Wang, W., Yang J. and Yu, P., Mining Patterns in Long Sequential Data with Noise, *ACM SIGKDD Explorations Newsletter*, 2(2): 28-33, 2000.
25. Witten, I.H. and Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques* (Second Edition), Morgan Kaufmann, 2005.
26. Yang, J., Wang, P. and Yu, P., Mining asynchronous periodic patterns in time series data. *In Proceedings of ACM SIGKDD*, pages 275-279, 2000.
27. Yemini, Y., Yemini, S. and Kliger, S., Apparatus and Method for Event Correlation and Problem Reporting, United States Patents and Trademarks Office, Patent Nos. 7003433, 6868367, 6249755 and 5528516, 2008.