University of Helsinki
Department of Computer Science
Series of Publications C, No. C-1996-7

Rule Discovery in Alarm Databases

Kimmo Hätönen, Mika Klemettinen, Heikki Mannila,
Pirjo Ronkainen, Hannu Toivonen

Helsinki, March 1996

Department of Computer Science
P. O. Box 26 (Teollisuuskatu 23)
FIN-00014 University of Helsinki, FINLAND

Rule Discovery in Alarm Databases

Kimmo Hätönen, Mika Klemettinen, Heikki Mannila,
Pirjo Ronkainen, Hannu Toivonen

University of Helsinki, Department of Computer Science
P. O. Box 26 (Teollisuuskatu 23)
FIN-00014 University of Helsinki, FINLAND
E-mail: mannila@cs.Helsinki.FI
Report C-1996-7, March 1996, 19 pages

Abstract
Telecommunication networks produce large amounts of alarm information daily. This data contains potentially valuable knowledge about the network. We present a methodology for the analysis of large telecommunication networks alarm databases. The methods used aim at discovering useful knowledge about the network which can be employed in real time alarm handling software for filtering uninformative alarms, for correlating alarms to construct hypotheses of faults, or for fault prediction.

The methodology is based on novel knowledge discovery methods for discovering patterns in alarm databases. We have implemented our methodology in the TASA (Telecommunication Network Alarm Sequence Analyzer) system which discovers patterns in alarm databases and provides tools for interactive identification of the interesting patterns.

Computing Reviews Categories and Subject Descriptors:

C.2.3	[Computer Systems Organization]: Computer-Communication Networks – Network Operations
H.3.3	[Information Systems]: Information Storage and Retrieval – Information Search and Retrieval
H.5.2	[Information Systems]: Information Interfaces and Presentation – User Interfaces
I.2.6	[Computing Methodologies]: Artificial Intelligence – Learning

General Terms:
Algorithms, Human Factors, Management.

Additional Key Words and Phrases:
Telecommunication Network Alarms, Data Mining, Knowledge Discovery, Discovery Methodology, Episodes, Association Rules, Alarm Correlation.

1 Introduction

Telecommunication networks of today produce large amounts of alarms daily. Fully employing this valuable data in network management is difficult, however, due to the high volumes of the fragmented information. In this paper, we will present a methodology for the analysis of large alarm databases, with the goal of discovering useful knowledge about the network. We present methods for automatic discovery of patterns in alarm databases and tools for interactive identification of the interesting patterns.

Alarm surveillance is a difficult network management task for several reasons (Figure 1). Networks can be large, with diverse elements and alarms, and networks themselves are different. Changes in the network happen frequently: elements are added, modified, or replaced, and the network configuration also changes. Alarms tend to occur in bursts, making fast response more important but more difficult. These factors imply that there typically are no human experts who would understand all the characteristics of alarm data.

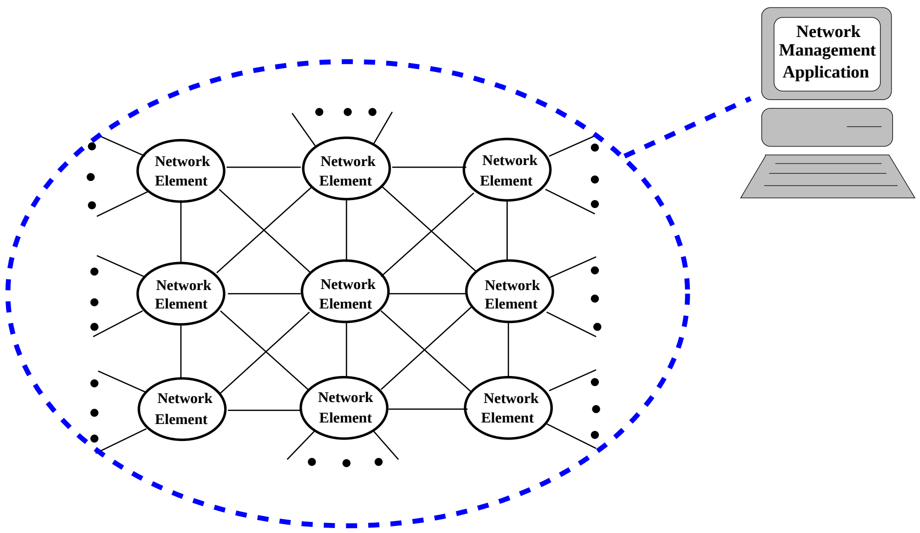

Figure 1: Schematic view of telecommunication network management.

As the flow of alarm data is dense, there is a user-interface problem in designing alarm handling software: how to show the data to the users so that the appropriate decisions can be made.

To alleviate the risk of overlooking important alarms or misinterpreting them, *filtering* and *correlation* methods are used to reduce the number of messages shown and to improve their information content (see, e.g., [6, 7, 9, 10, 11, 18]). For this one needs knowledge about the alarms and their interrelationships, and filtering and correlation systems are often implemented as expert systems with carefully hand-crafted knowledge bases. The required knowledge can in principle be obtained from the designers of individual components or from experienced operators, but this process can be tedious. In this article we consider the use of knowledge discovery methods to automatically find useful patterns in the alarms.

Knowledge discovery in databases (KDD), also called data mining, has recently attracted a lot of interest in the database community (for overviews, see [5, 19]). KDD can be loosely defined as the task of obtaining useful and interesting knowledge from large collections of data. It combines methods and tools from machine learning, statistics, and databases. We use novel KDD methods to discover effectively very large sets of rules from alarm databases. We then use iterative information retrieval methods to examine interactively the discovered rules and to find the most interesting and useful ones. A related algorithm for discovering rules from alarms has been presented in [8].

Patterns discovered in an alarm database can be useful in several ways. They can be employed in real time alarm handling software for filtering uninformative alarms, for correlating alarms to construct hypothesis of faults, or for prediction of faults. The discovered patterns can also help to reveal anomalies in the behaviour of the network. Similar approaches have been used successfully in process control tasks [17].

The ideas expressed in this article have been implemented in the TASA (Telecommunication Network Alarm Sequence Analyzer) system for discovering knowledge from large alarm databases. TASA is in prototype use, and the results are encouraging: rules found by TASA have been integrated to the alarm handling software of telecommunication operators.

The rest of this article is organized as follows. Section 2 introduces our knowledge discovery process and discusses the choice of knowledge representation. In Section 3 we give our exact rule formalisms. We then present in more detail three steps of the KDD process: data collection and preprocessing are discussed in Section 4, discovery methods are presented in Section 5, and Section 6 describes methods for managing large sets of discovered rules. Section 7 is a short conclusion.

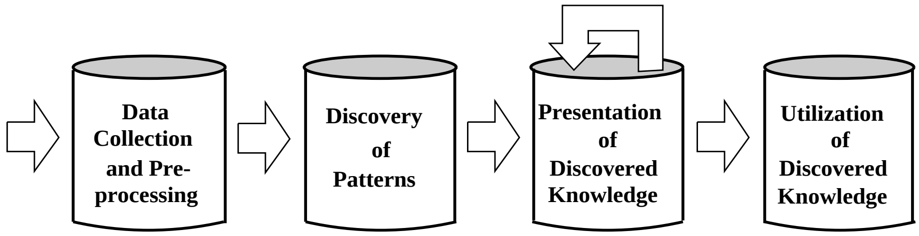

Figure 2: Knowledge discovery process.

2 Knowledge discovery process

The KDD process includes steps from data collection and pre-processing to putting the discovered knowledge into use. In the following we will shortly present the process model we use.

The tasks in the process are the following (see Figure 2).

1. Data collection and pre-processing: obtaining the data, identifying and handling of errors and missing data, transformations of attribute values, identification of necessary background knowledge.

2. Discovery of patterns (data mining).

3. Presentation of the discovered knowledge: interactive selection of interesting patterns, visualization of results, etc.

4. Putting the discovered knowledge into use.

The TASA system incorporates components for two parts of the KDD process: pattern discovery (or data mining) and presentation of the knowledge.

KDD is an iterative and interactive process [3, 4]. No realistic KDD system can be expected to discover useful knowledge without interaction with the user: knowing the interests and using the background knowledge of users is vital for succesful knowledge discovery. Also, iteration is essential: after receiving some knowledge the user is better able to focus the search to more interesting areas.

While it is typical for KDD systems to iterate between the discovery and presentation steps, our approach is different: we discover a very large collection of rules at once, and the iteration is performed mostly in the presentation step, avoiding repetitive runs of the computationally more demanding discovery step.

We now move on to discuss our choice of pattern types to be discovered. Different types of knowledge can be discovered from an alarm database, for example neural network models, hazard models, or rule-based representations. If the goal would be just to obtain good predictive performance, a neural network could be useful. However, a very important goal is the *comprehensibility* of the discovered knowledge: "black boxes" are largely undesired by telecommunication operators. This rules out the simple-minded use of neural networks.

Other possible methods include viewing an alarm sequence as a marked point process and using the hazard rate based methods of analysis of failure-time data [12]. Combined with the Bayesian paradigm, the statistical machinery for these types of models is very powerful, and elaborate simulation mechanisms exist for their analysis (e.g., the EM and Gibbs methods, see [21]). However, these methods require a lot of human effort in building the statistical models, as well as enormous amounts of computational resources. Currently they are not feasible for the analysis of hundreds of different types of alarms and their potential relations.[1]

We use rule-based formalisms for knowledge discovery in the TASA system. The general form of *episode rules* is the following:

> If a certain combination of alarms occurs within a time period, then an alarm of a given type will occur within a time period.

An example of such a rule is: "If alarms of types *link alarm* and *link failure* occur within 5 seconds, then an alarm of type *high fault rate* occurs within 60 seconds with probability 0.7." The general form of *association rules* is the following:

> If an alarm has certain properties, then it will have other given properties,

and an example association rule is "If an alarm is sent during office hours by an element of type *base station*, then the alarm has severity level 1 with probability 0.9."

The reasons for choosing these types of knowledge to be discovered are the following.

1. Comprehensibility: such rules are easy to understand.

[1]We are, however, experimenting on Markov Chain Monte Carlo methods for Bayesian analysis of alarm data [2].

Figure 3: Example sequence of predicates.

2. Characteristics of the domain: such rules can be representations of simple small causal relationships.

3. Existence of efficient algorithms: rules of the above form can be discovered efficiently even in alarm databases of networks with thousands of elements sending thousands of alarms a day.

3 Episode and association rules

In this section we will give the exact rule formalisms and discuss the rule types discovered.

Each alarm can be abstractly viewed as a quadruple (c, a, p, t), where c is the component that sent the alarm, a is the alarm type, p contains other (possibly derived) attributes of the alarm, and t is the time when the alarm occurred.

An *alarm predicate* is an expression that can be evaluated from a single occurrence of an alarm, e.g., "alarm type is *link failure*", "alarm occurred during office hours", or "alarm type is *link failure* or *link alarm*, and alarm severity class is 3".

Episodes An *episode* is a pair $\alpha = (V, \leq)$, where V is a collection of alarm predicates, and $\leq$ is a partial order on V. Given a sequence of alarms S, an episode $\alpha = (V, \leq)$ *occurs* within S if there is a way of satisfying the alarm predicates in V using the alarms of S so that the partial order $\leq$ is respected. Given a window width W, the *frequency* of an episode α is the fraction of windows of width W in S in which α occurs. Intuitively, episodes consist of alarms that have certain properties and occur in a certain partial order. The higher the frequency of an episode, the more often it occurs in the database.

The most common types of partial orders used are (1) total orders, i.e., the predicates of each episode have a fixed order; such episodes are called *serial*, and (2) trivial partial orders, where the order is not significant at all; such episodes are called *parallel*. Figure 3 presents an example sequence of alarm predicates $A, B, \ldots$. The serial episode consisting of predicates E and

F, i.e., of E and F in the fixed order, occurs several times in the sequence. The parallel episode of predicates A and B, i.e., A and B in either order, occurs also several times.

An *episode rule* $r = (\alpha, e, W, W')$ consists of an episode $\alpha = (V, \leq)$, a distinguished predicate e of V, and two window widths W and W'. The interpretation of r is the following: the predicates in $V' = V \setminus \{e\}$ and the restriction of $\leq$ to V' define the left-hand side of the rule, and e is the right-hand side of the rule. The rule states: if alarms satisfying V' occur in the right order within W time units, then also e occurs in the location described by $\leq$, within W' time units (i.e., the whole episode α occurs). The time interval W may not be larger than W'.

Given a sequence of alarms S, the *confidence* of a rule $r = (\alpha, e, W, W')$ is t'/t, where t is the frequency of $\alpha \setminus \{e\}$ with respect to window width W, and t' is the frequency of α with respect to window width W'. The *frequency* of a rule is t'. An episode rule, in effect, gives the conditional probability that a certain alarm occurs in a window given that a certain combination of alarms occurs.

Examples of episode rule types discovered in the TASA system are:

- If an alarm of type *link unavailable* occurs, then an alarm of type *route unavailable* occurs within 5 seconds with probability 1.0.
- If alarms of types *link alarm* and *link failure* occur within 5 seconds, then an alarm of type *high fault rate* occurs within 60 seconds with probability 0.7.
- If an alarm of type *self reset* precedes an alarm of type *restart failure*, and *mains failure* precedes *battery low*, all within 1 minute, then *link unavailable* will follow within 4 minutes with probability 0.9.

These types of rules describe patterns of several alarms in time. They suggest groups of alarms that are potentially causally related, and also indicate their temporal relation. Given such rules that hold in an alarm database, a domain expert is likely to be able to verify the rules. Some of the rules may be based on known causal connections, some may be irrelevant — and some rules may give new insight to the behaviour of the network elements.

Association rules An *association rule* is an expression $X \Rightarrow Y$, where X and Y are sets of predicates, the left-hand and right-hand sides of the rule.

Given a set of alarms, the *confidence* of such a rule is the conditional probability with which predicates in Y are satisfied by an alarm in the database given that predicates in X are satisfied by the alarm. In other words, association rules describe which properties of alarms occur together, and with which probability. The *frequency* of an association rule is the fraction of alarms that satisfy the rule.

Examples of association rule types found by TASA are:

- If an alarm is of type *link alarm*, then it was sent by the network element X with probability 0.5.
- If an alarm is sent during office hours by an element of type *base station*, then the alarm has severity level 1 with probability 0.9.

Such rules give an overview of what the alarms in the database are like. A comparison with association rules from last week, last month, etc. could point out trends and changes. Unusually high confidences may point to potential anomalies in the network. Note that association rules do not take the temporal relationships of the alarms into account.

Parameters In the pattern discovery phase, *all* episode and association rules whose frequency exceeds a user-specified threshold are output. While the idea is to perform rather unfocused search for rules, the user has to give some parameters. The less the parameters restrict the rules to be found, the more material there is for the rule presentation phase. The parameters are as follows.

- First of all, the alarm predicates are given by the user. For episode rules, the type of the alarm and the network element that sent the alarm are the most typical predicates. For association rules in turn, much larger predicate classes are useful. We discuss the choise of predicates in the next section in conjuction with data collection and pre-processing.
- With episodes, the user specifies the *class of discovered rules* by defining what types of partial orders are allowed in episodes, what types of alarm predicates are used, and what are the values for the two window widths W and W'. Serial and parallel episodes are the most common types; it is not quite clear how useful truly partially ordered (i.e., non-serial and non-parallel) episodes are in the applications.

- For episode rules, the user supplies two window widths with which the rules are constructed. In our experiments, the application experts have preferred time windows ranging from 5 seconds to 10 minutes.
- For both episode and association rules, the frequency threshold is given by the user. If the threshold is low, also rules whose left-hand side occurs rarely are included in the output. With lower thresholds the number of rules is thus larger, and the discovery time is longer.

For each episode rule and association rule satisfying the given parameters, TASA gives the confidence, frequency, and statistical significance.

4 Data collection

The first step in the KDD process is collecting the data. Alarm data is readily available in alarm databases, and the data is usually of high quality. This is very fortunate; in some domains this step can take up to 80 % of the total time needed for the discovery process.

Some problems still remain. One is related to the time-dependent nature of the interesting events in the network. The time field of alarm messages is not very reliable: there can be differences of several minutes in the synchronization of the clocks. As our goal is to locate regularities that are intimately connected to the temporal aspects of alarms, such errors can cause problems. A solution is to perform the analysis on a coarse granularity.

An important pre-processing step is the selection of alarm predicates. The first step is to identify useless fields in alarm messages, such as those containing redundant information (alarm text is redundant, if it can be detemined from the alarm type) or irrelevant information (for example, the physical location of the alarming element in a rack might be considered irrelevant). Alarm predicates can be derived from the fields considered interesting by constructing boolean-valued assertions. In addition to straight-forward derivation of predicates such as "alarm type = *link fault*", "alarm type = *link failure*", etc., for instance classification predicates of the form "alarm type = *link fault* or *link failure*" can be derived. Or, it can be useful to derive from date and time fields predicates such as "office hours = yes" and "office hours = no", "weekday = Mon", etc.

The sets of predicates used for episode rules and association rules are different. In episode rules, it is typically the alarm type and the alarming

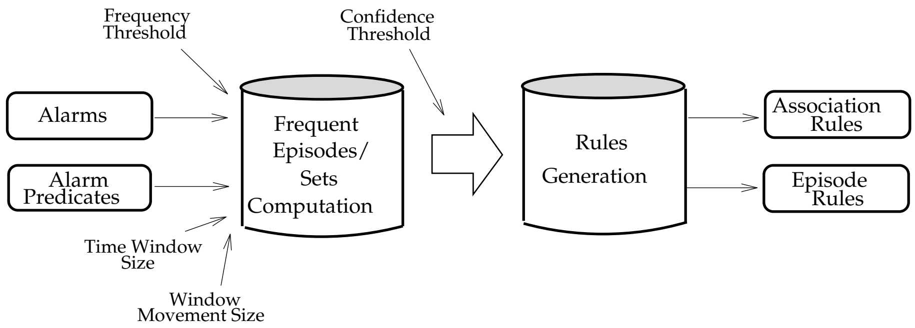

Figure 4: Rule discovery in TASA.

network element that are of interest. In association rules, in turn, a large variety of predicates can be useful.

5 Discovering all rules

In this section we outline the algorithms that are used in TASA to discover all rules satisfying the user-given conditions. The algorithm for discovering episodes is presented in more detail in [16] for a slightly simpler setting; the association rule algorithm is presented in more detail, e.g., in [1, 15].

The episode rule discovery task in the TASA system can be formalized as follows (see Figure 4). Given a sequence S of alarms, a set E of alarm predicates, a class $\mathcal{E}$ of episodes built from the predicates of E, a frequency threshold c, and window widths W and W', find the confidences of all rules (α, e, W, W') whose frequency is at least c. For association rule discovery, an alarm database S, a set E of alarm predicates, and a frequency threshold c are given, and the task is to find the confidences of all rules $X \Rightarrow Y$ whose frequency is at least c.

The main algorithms for the two tasks are based on the same basic idea; our focus will be on the more complex task of discovering episode rules. We first discuss how the frequently occurring episodes (or predicate sets for association rules) are computed; this is the computationally harder part of the problem. We then show how rules are obtained from this information.

Computing frequent episodes or predicate sets The problem we consider in this subsection is the following. Given S, E, $\mathcal{E}$, W, and c as above, find the frequencies of all episodes from $\mathcal{E}$ such that the frequency is at least

c. The following algorithm will output all such episodes [16].

1. $C_1 := \{\{e\} \mid e \in E\}$;
2. $i := 1$;
3. **while** $C_i \neq \emptyset$ **do**
4. *recognition*: read the sequence S, and let L_i be the episodes of C_i that occur often enough with respect to W and c;
5. *building*: compute C_{i+1} to contain those episodes of $i+1$ alarm predicates whose all subepisodes occur often enough;
6. $i := i + 1$;
7. **od**;
8. **for** all i, output L_i;

The algorithm starts with simple (i.e., general) episodes and proceeds to larger episodes. The algorithm works iteratively: first a candidate collection C_i is generated, candidates are counted in the alarm sequence, those occurring often enough are saved in the collection L_i, and finally new candidates are generated again. Essential for the efficiency of the algorithm is the following observation. If an episode does not occur often enough, then its superepisodes — which are more specific — cannot occur often enough. Therefore, the candidate collection C_i of episodes is built (Step 5) to contain only episodes whose *all* subepisodes occur often enough. For more details, see [16].

To find frequent predicate sets, essentially the same algorithm and the same observation hold [15]. Candidates are now sets of predicates, and instead of considering the alarm database as a sequence, it is seen as a set of alarms. In Step 4, "recognition", for each candidate set the number of satisfying alarms is computed and compared to the frequency threshold. In Step 5, "building", new candidate sets are constructed such that *every* subset of a candidate set is frequent. For more details, see [15].

We have analyzed with these methods alarm databases from different fixed and cellular networks. Running times on a Pentium-based PC range from few seconds to an hour, depending on the database and the parameters. Parallel episodes with alarm type as the only predicate can be discovered in a sequence of 70,000 alarms with a window width of 10 seconds in few seconds, whereas finding frequent predicate sets in a database of 40,000 alarms with over 8,000 predicates can take an hour.

Rule generation Once we know how often each episode from a class of episodes occurs in an alarm sequence, finding the confidences of rules is easy.

For a potential rule (α, e, W, W') we have that the frequencies of the episode $\alpha \setminus \{e\}$ for time window W and of the episode α for time window W' exceed the threshold; the former we have computed in Step 4 of the algorithm. Now the potentially holding rules can be generated efficiently from the frequent episodes. Then the task of computing rule confidences can be done in a single pass through the data.

For association rules this is even simpler. Given a frequent predicate set X, for all its subsets $Y \subset X$ we have that the confidence of the rule $(X \setminus Y) \Rightarrow Y$ is t'/t, where t is the frequency of $X \setminus Y$ and t' is the frequency of Y. Since these frequencies have already been computed in Step 4, the database is not needed for rule generation at all.

Rule generation is fast and the time spent for it is neglible, compared to the time used to find the frequent episodes of predicate sets. The amount of rules generated can be very large, especially with low frequency thresholds. Even with reasonable parameters the number of rules has ranged from about 1,000 to over 20,000.

6 Rule presentation

We now move to the interactive step in our KDD process, the handling of large collections of discovered rules. As stated earlier, our approach is to discover large collections of rules with the algorithms presented in the previous section, and concentrate the interaction and iteration to the phase where the results are represented to the user. The advantage is, in short, that it is very fast to change the focus or viewpoint when the rules are already available. Of course, any restrictions placed for the discovery algorithms, such as the frequency threshold, cannot be relaxed without going back to the rule discovery phase.

The aim of knowledge discovery in databases is to extract *interesting* knowledge from large collections of data. What is interesting varies from one situation to another. The interestingness criteria — such as what is on the right-hand sides of the rules, or how significant the rules should be — are in many KDD systems given as inputs to the pattern extraction process [14, 20]. If the user wants to change his or her point of view, the data has to be analyzed anew. This can require a lot of computational effort and cause delays in the KDD process. In the TASA system we aim at a KDD process where going back to the actual data for a new pattern extraction phase has to be done only seldom.

Antecedent	Consequent	Conf	Frequency	Sign
1234_44545	1234_66656	0.59	0.0000157	1.00
1234_44545	1234_11095	0.58	0.0000152	0.99
1234_44545	6789_44545	1.00	0.0000262	0.88
1234_44545	6789_66656	0.59	0.0000157	0.97
1234_44545	6789_11095	0.58	0.0000152	0.99
1234_44545	3245_44545	0.72	0.0000189	0.79
1234_44545	3245_66656	0.35	0.0000094	–
1234_44545	3245_11095	0.34	0.0000089	–
1234_31608	6789_31608	0.77	0.0000367	0.99
1234_31608	3245_31608	0.69	0.0000330	0.88
1234_66656	1234_44545	0.59	0.0000157	0.99
1234_66656	1234_11095	0.58	0.0000152	0.99
1234_66656	6789_44545	0.59	0.0000157	0.89
1234_66656	6789_66656	1.00	0.0000262	0.99
1234_66656	6789_11095	0.58	0.0000152	0.97
1234_66656	3245_44545	0.35	0.0000094	–

Figure 5: Simple episode rules generated with TASA.

Rule retrieval The TASA system offers a variety of selection and ordering criteria for rules and supports iterative retrieval from the discovered knowledge. We have identified the following types of operations the users want to do with the rule sets:

1. *Pruning*: removal of uninteresting rules or selection of interesting ones.
2. *Ordering* of rules according to some criteria.
3. *Grouping*: clustering of rules into groups of rules that have similar effects in the analyzed data sets.

Pruning and ordering can be done according to the alarm predicates in the rules, e.g, types or severities of the alarms, the sending network element, etc., or according to the confidence, frequency, or statistical significance of the rules.

The point of focus can be changed by selecting or removing rules with *templates* [13]. Templates are simple regular expressions that describe, in

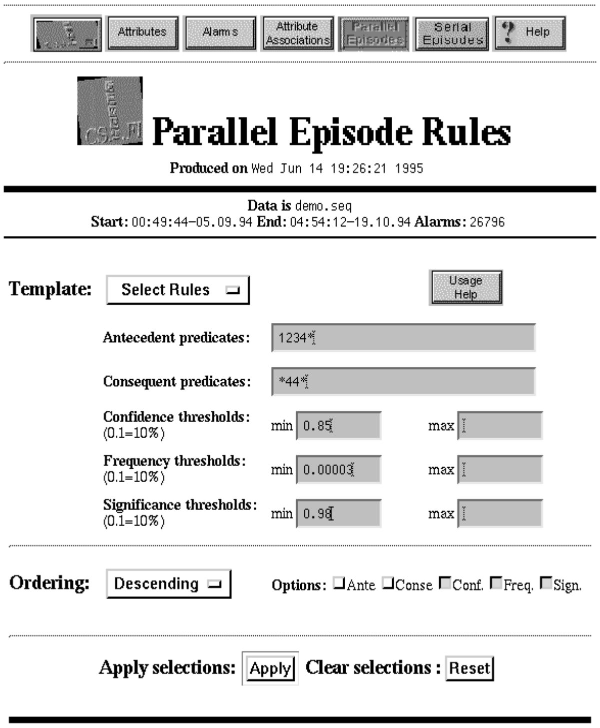

Figure 6: Fill-out form for pruning and selecting rules.

terms of alarm predicates, the form of rules that are to be selected (inclusive template) or removed (restrictive template). This simple technique is surprisingly powerful. For example, episode rules about alarms from separate subsystems can be obtained by using templates that reject all rules where the alarming network elements are in the same subsystem. Or, focus can be set to day-time alarms by selecting only association rules that contain the predicate "office hours = yes".

Clustering of rules aims at giving a larger picture of the behavior of the alarm sequence. In the data there are often various explanations for the occurrence of a particular alarm type, say *link unavailable*. Clustering methods can be used to assign rules to groups so that two rules with right-hand side *link unavailable* belong to the same cluster if they often explain or predict *link unavailable* in similar situations [22]. This can be useful in pointing out potentially related rules.

For visualization the TASA system offers some simple facilities. E.g., an interesting picture of the interaction between the left and right-hand sides of

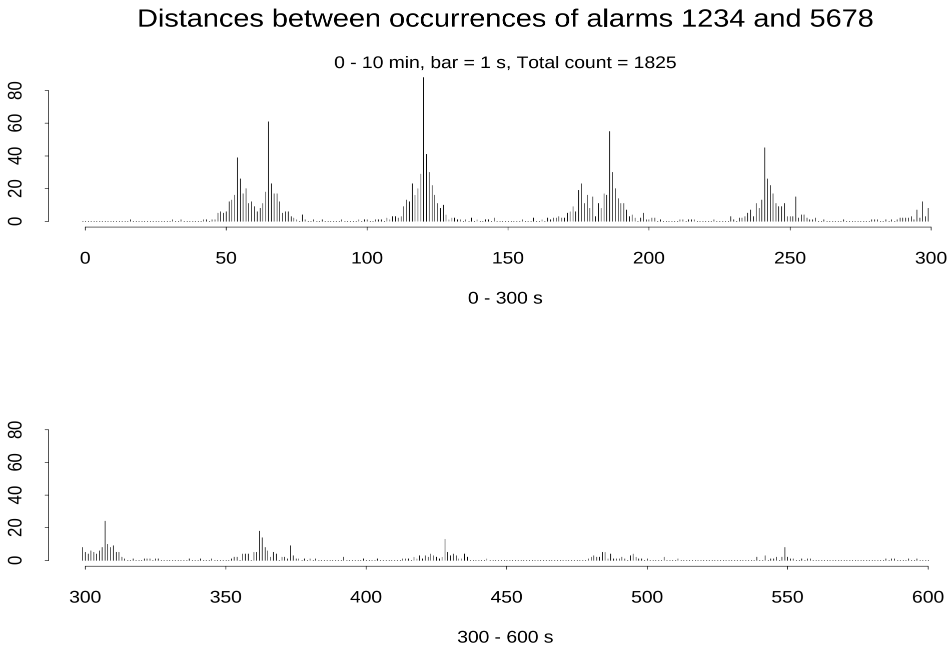

Figure 7: Distance histogram for the left-hand and right-hand sides of a rule.

a simple episode rule can be obtained by drawing a histogram showing the distance from each occurrence of the left-hand side to the nearest occurrence of the right-hand side. Such histograms are valuable guides for locating possible periodic relationships between the left and right-hand sides, as is demonstrated by Figure 7. Note that the episode rule formalism does not capture the details of such relationships.

User interface For the user interface system we have adopted the method used in [20]: we have based the implementation of the rule browsing system on the use of the HTML language and standard WWW browsers for HTML documents. This gives several immediate benefits, both in terms of added functionality and in terms of ease of implementation. This architecture also means that it is extremely simple to port the rule browsing system to different platforms.

The structure of the analysis documentation is the following. The Home Page of an analyzed data set gives a brief overview of the data and contains links to the results of the analysis. Each rule set is an HTML document (e.g., Figure 5). Hypertext links lead from predicates in the rules to descriptions of the predicates: e.g., clicking on an alarm type in an episode rule takes the user to a page containing the alarm text, the total number of alarms of the type, etc. Selection, ordering, and grouping criteria for a rule set are given with an HTML fill-out form (Figure 6). On the form the user can define rule templates, and give minimum and maximum values for rule confidence, frequency, and statistical significance. Simple ordering criteria for the resulting rules can be given. The operations are applied on the rule set, and a new HTML document is created "on-the-fly". The user can move freely in the history path and reformulate the selections from any of the former pages. If the results are interesting, the user can save the on-screen page for further examination. The original data is also linked to the system, so that the user is able to see how the rules are actually supported by it.

Example As an example of how the system can be used, consider the following scenario. Assume you have discovered association rules for the current year. First, you might want to see what the alarms have been like in the current month, say June, so you use a template to select rules with the predicate "month = June" as the antecedent.

The number of selected rules is still very large. You decide to restrict the rule right-hand side to only contain one predicate, and you order the rules by their confidences.

Looking at the selected rules, you see the rule "if month = June then alarm type = *connection failure*" with confidence 0.12, and you infer that an unusually large fraction of alarms in June has been of type *connection failure*. To see in more detail what the alarms have been like, you refine your template and select rules with "month = June and alarm type = *connection failure*" as the antecedent.

Looking at the new set of selected rules, you see that a lot of rules concern the network element X. That reminds you of maintenance undertaken during last week that explains those rules. To remove the rules, you apply a restrictive template with the predicate "network element = X".

The resulting set of rules shows nothing special, but just to make sure you want to compare the rules with the corresponding rules from May. You open a copy of the window, and change the first template to "month = May". If there is anything special or interesting, the viewing criteria can be refined or altered again.

7 Conclusions and future work

We have described a new methodology for analyzing large telecommunication alarm databases. Our approach is based on a knowledge discovery process; we presented our model of the process and methods for discovering and managing large collections of rules that hold in the analyzed alarm database. Discovered rules can be useful in real time alarm filtering and correlation; they also have potential for fault prediction and anomaly detection.

Our experiences with the methodology are encouraging. The rules found by TASA are intelligible to the network management experts. Some of the rules are deemed trivial, while others are considered highly interesting. We have discovered with TASA new rules that have actually been integrated to network surveillance systems.

In the future, we would like to extend the methodology to handle measurement data as well. For instance, the view of the network given by call or traffic volume time series is quite different from the problem-oriented view given by an alarm database. In particular, analyzing alarm databases and measurement databases in combination could open interesting views to telecommunication network management.

References

[1] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. Fast discovery of association rules. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307 – 328. AAAI Press, Menlo Park, CA, 1996.

[2] Elja Arjas, Heikki Mannila, Marko Salmenkivi, Riikka Suramo, and Hannu Toivonen. BASS: Bayesian analysis of event sequences. In *XII Symposium on Computational Statistics (COMPSTAT'96)*, Barcelona, Spain, 1996. To appear.

[3] Ronald J. Brachman, Peter G. Selfridge, Loren G. Terveen, Boris Altman, Alex Borgida, Fern Halper, Thomas Kirk, Alan Lazar, Doborah L. McGuinness, and Lori Alperin Resnick. Integrated support for data archaeology. In *Knowledge Discovery in Databases, Papers from the 1993 AAAI Workshop (KDD'93)*, pages 197 – 211, Washington, D.C., July 1993.

[4] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery: An overview. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 1 –34. AAAI Press, Menlo Park, CA, 1996.

[5] Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining.* AAAI Press, Menlo Park, CA, 1996.

[6] J. Goldman, P. Hong, C. Jeromnimon, G. Louit, J. Min, and P. Sen. Integrated fault management in interconnected networks. In B. Meandzija and J. Westcott, editors, *Integrated Network Management, I*, pages 333 – 344. Elsevier, 1989.

[7] R. M. Goodman, B. E. Ambrose, H. W. Latin, and C. T. Ulmer. Noaa – an expert system managing the telephone network. In *Integrated Network Management IV*, pages 316 – 327. Chapman & Hall, London, 1995.

[8] R. M. Goodman and H. Latin. Automated knowledge acquisition from network management databases. In I. Krishnan and W. Zimmer, editors, *Integrated Network Management, II*, pages 541 – 549. Elsevier Science Publishers B.V (North-Holland), Amsterdam, 1991.

[9] Gabriel Jakobson and Mark Weissman. Real-time telecommunication network management: Extending event correlation with temporal constraints. In *Integrated Network Management IV*, pages 290 – 301. Chapman & Hall, London, 1995.

[10] Gabriel Jakobson and Mark D. Weissman. Alarm correlation. *IEEE Network*, 7(6):52 – 59, November 1993.

[11] J. F. Jordaan and M. E. Paterok. Event correlation in heterogenous networks using OSI management framework. In H.-G. Hegering and Y. Yemini, editors, *Integrated Network Management, III*, pages 683 – 695. Elsevier Science Publishers B.V (North-Holland), Amsterdam, 1993.

[12] J. D. Kalbfleisch and R. L. Prentice. *The Statistical Analysis of Failure Time Data.* John Wiley Inc., New York, NY, 1980.

[13] Mika Klemettinen, Heikki Mannila, Pirjo Ronkainen, Hannu Toivonen, and A. Inkeri Verkamo. Finding interesting rules from large sets of discovered association rules. In *Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94)*, pages 401 – 407, Gaithersburg, MD, November 1994. ACM.

[14] Willi Klösgen. Efficient discovery of interesting statements in databases. *Journal of Intelligent Information Systems*, 4(1):53 – 69, 1995.

[15] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Efficient algorithms for discovering association rules. In Usama M. Fayyad and Ramasamy Uthurusamy, editors, *Knowledge Discovery in Databases, Papers from the 1994 AAAI Workshop (KDD'94)*, pages 181 – 192, Seattle, Washington, July 1994.

[16] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Discovering frequent episodes in sequences. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD'95)*, pages 210 – 215, Montreal, Canada, August 1995.

[17] R. Milne, C. Nicol, M. Ghallab, L. Trave-Massuyes, K. Bousson, C. Dousson, J. Quevedo, J. Aguilar, and A. Guasch. TIGER: real-time situation assessment of dynamic systems. *Intelligent Systems Engineering*, (Autumn):103 – 124, 1994.

[18] Y. A. Nygate. Event correlation using rule and object based techniques. In *Integrated Network Management IV*, pages 278 – 289. Chapman & Hall, London, 1995.

[19] Gregory Piatetsky-Shapiro and William J. Frawley, editors. *Knowledge Discovery in Databases*. AAAI Press, Menlo Park, CA, 1991.

[20] Gregory Piatetsky-Shapiro and Christopher J. Matheus. The interestingness of deviations. In Usama M. Fayyad and Ramasamy Uthurusamy, editors, *Knowledge Discovery in Databases, Papers from the 1994 AAAI Workshop (KDD'94)*, pages 25 – 36, Seattle, Washington, July 1994.

[21] Martin A. Tanner. *Tools for Statistical Inference — Methods for the Exploration of Posterior Distributions and Likelihood Functions*. Springer-Verlag, Berlin, 1993.

[22] Hannu Toivonen, Mika Klemettinen, Pirjo Ronkainen, Kimmo Hätönen, and Heikki Mannila. Pruning and grouping of discovered association rules. In *Workshop Notes of the ECML-95 Workshop on Statistics, Machine Learning, and Knowledge Discovery in Databases*, pages 47 – 52, Heraklion, Crete, Greece, April 1995.