ELSEVIER

Information and Software Technology 41 (1999) 557–567

INFORMATION AND SOFTWARE TECHNOLOGY

# Interactive exploration of interesting findings in the Telecommunication Network Alarm Sequence Analyzer TASA

M. Klemettinen*, H. Mannila[1], H. Toivonen[2]

*University of Helsinki, Department of Computer Science, P.O. Box 26, University of Helsinki, FIN-00014 Helsinki, Finland*

**Abstract**

In this paper we describe the final version of a knowledge discovery system, Telecommunication Network Alarm Sequence Analyzer (TASA), for telecommunication networks alarm data analysis. The system is based on the discovery of recurrent, temporal patterns of alarms in databases; these patterns, episode rules, can be used in the construction of real-time alarm correlation systems. Also association rules are used for identifying relationships between alarm properties. TASA uses a methodology for knowledge discovery in databases (KDD) where one first discovers large collections of patterns at once, and then performs interactive retrievals from the collection of patterns. The proposed methodology suits very well such KDD formalisms as association and episode rules, where large collections of potentially interesting rules can be found efficiently. When searching for the most interesting rules, simple threshold-like restrictions, such as rule frequency and confidence may satisfy a large number of rules. In TASA, this problem can be alleviated by templates and pattern expressions that describe the form of rules that are to be selected or rejected. Using templates the user can flexibly specify the focus of interest, and also iteratively refine it. Different versions of TASA have been in prototype use in four telecommunication companies since the beginning of 1995. TASA has been found useful in, e.g. finding long-term, rather frequently occurring dependencies, creating an overview of a short-term alarm sequence, and evaluating the alarm data base consistency and correctness. © 1999 Elsevier Science B.V. All rights reserved.

*Keywords:* Discovery system; Telecommunications; Knowledge discovery methodology; Episode rules; Association rules

## 1. Introduction

The goal of knowledge discovery (KDD) is to find useful patterns in data. Success in this task requires not only that the KDD tools and the knowledge representation formalisms are such that useful patterns can be found and comprehended, but also that the user understands what it means that certain patterns are output while certain others are not.

In this paper we present a knowledge discovery methodology where the user has a total, flexible control over the output of discoveries. We describe the final version of a knowledge discovery system, Telecommunication Network Alarm Sequence Analyzer (TASA) [1], that supports the methodology. TASA was built at the University of Helsinki, in cooperation with four telecommunication companies, and it has recently been successfully fielded. The pattern types discovered by TASA are episodes [2] and association rules [3].

The problem of locating a small set of truly interesting information is a generic problem in data mining (see, e.g. [4]): while formal statistical criteria for strength and significance of the discovered items abound, it is much harder to know which parts of the discovered knowledge really interest the user. Concepts like actionability and unexpectedness [5], as well as novelty, usefulness, simplicity and generality [6], are all more or less subjective and data-dependent measures. Many of them require more background knowledge than is usually available in the beginning of a knowledge discovery process. That is why TASA assists the user through the discovery process and makes him familiar with both the data and the findings before he needs to start expressing his subjective criteria for interestingness.

Most KDD systems prune and rank the set of patterns automatically, typically aiming at a small, non-redundant collection of the most interesting (e.g. strong, useful, surprising, or valuable) patterns. We adopt a different approach. Our methodology emphasizes the following two characteristics:

1. A large, unfocused collection of potentially interesting patterns is discovered at once.
2. The focus on the discovered patterns can be set iteratively and interactively.

As a result of the pattern discovery phase of our KDD

* Corresponding author.

[1] Currently affiliated to Microsoft Research.

[2] Currently affiliated to Rolf Nevanlinna Institute, University of Helsinki.

0950-5849/99/$ - see front matter © 1999 Elsevier Science B.V. All rights reserved.
PII: S0950-5849(99)00019-1

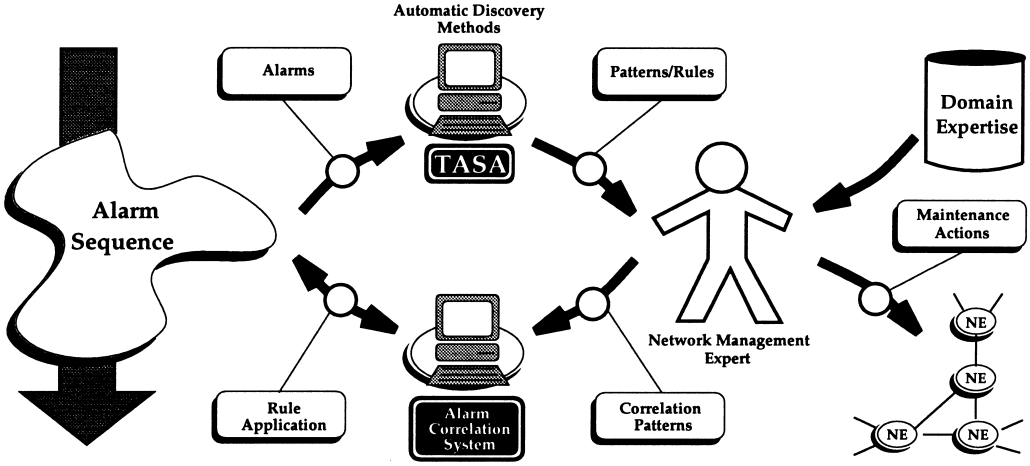


Fig. 1. An alarm correlation model using discovered alarm patterns.

process model, all patterns considered potentially interesting are produced. All the patterns are, however, not supposed to be interesting to a particular user or in a particular situation. On the contrary a major feature of TASA is to assist in the exploration of the discovered patterns.

TASA supports interactively focused views to the rules with different pruning, ranking, and structuring criteria. Sometimes considerable amounts of rules remain, even when the user has found the desired focus with the described methods. Automatic pruning, sorting, and structuring methods are at this point available for invocation by the user, especially for the removal of redundancy.

One of the focal points of this paper is the use of templates [7] in the interactive exploration of interesting findings. Templates are pattern expressions that describe the forms of rules that are to be selected or rejected. In the context of association and episode rules, templates define a set of rules by specifying what attributes occur in the antecedent and what in the consequent. Templates are a powerful formalism for the subjective pruning of a rule set and they can be implemented very efficiently after all rules have already been discovered.

The paper is organized as follows. Section 2 gives an overview of the problem domain as well as the role and the use of TASA. Then, in Section 3, we briefly review the characteristics of the discovery algorithms used in TASA and show how they are applied in the system. Section 4 concentrates on the exploration methodology and gives a number of examples. We review related work in Section 5. Finally, Section 6 concludes the paper by discussing the system usability and experiences.

## 2. TASA system overview

In Section 2.1, we first look at the environment in which TASA is used; a more detailed description of the domain area, telecommunication networks alarm databases, can be found in Ref. [8]. Then, in Section 2.2, we discuss the main methodology behind the TASA system. In Section 2.3, we briefly present the structure and main functions of the TASA system.

### 2.1. Telecommunication networks and TASA

Faults in a telecommunication network are reported to *operations and maintenance centers* in the form of *alarms*, messages emitted by network elements, typically when a problem is encountered. The goal of *alarm correlation* (see Ref. [9]) is to reduce the amount of information shown to the network managers, improve the usefulness of the information, identify the most probable faults that caused the alarms and possibly even propose corrective actions. *Alarm correlation systems* often are rule-based expert systems that remove redundant alarms, filter out low-priority alarms and substitute a set of alarms by a more informative message if possible. Unfortunately, building an alarm correlation system is a difficult task. Networks are large and network elements are complex. The number of correlation patterns needed in the system specification can be very large, and acquiring the patterns from technical experts is a tedious task.

A *correlation pattern* describes a situation that can be recognized in an alarm sequence and further acted on. Typically, a correlation pattern is an expression on the set of active alarms of, e.g. the last 5 min. Associated with each correlation pattern is a *correlation action*, which is to be executed when the corresponding pattern occurs. The correlation action takes care of, e.g. filtering the alarms.

TASA is a data mining tool for analyzing telecommunication networks alarms. The purpose of TASA is to aid in the knowledge acquisition phase for creating alarm correlation model, and to give new views to the alarms (See Fig. 1). First, a large database of alarms is analyzed off-line, and both temporal connections between different types of alarms and relationships between alarm properties are discovered

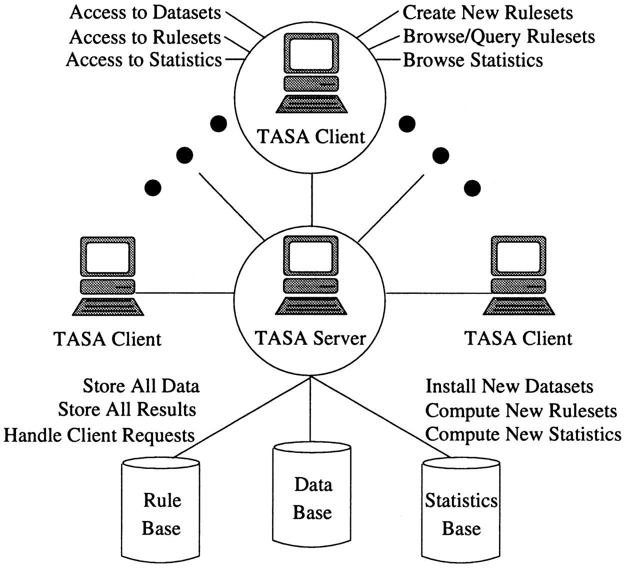


Fig. 2. Now TASA system structure based on client/server architecture.

automatically. Then, the network management specialists analyze the rules found and based on available background knowledge and knowledge inferred from the rules, select interesting ones. Finally, the selected rules are converted into correlation rules and are applied in real-time fault identification.

In TASA, we consider two kinds of recurrent patterns, episode rules [10] and association rules [3]. Episode rules describe temporal proximity and temporal ordering of recurrent combinations of alarms in a given alarm database, and they can be used as the basis for correlation patterns. Association rules describe, in turn, the properties of individual alarms without taking the temporal relationships of the alarms into account.

### 2.2. TASA methodology

A KDD process, adapted from [11], consists of:

1. Data preprocessing (selection, cleaning, etc.).
2. Data transformation and input selection for discovery phase.
3. Discovery of patterns.
4. Presentation of the results.
5. Interpretation and utilization of the results.

We follow the general framework, but there are two important characteristics that separate the methodology from the others [1,12]:

1. In the rule discovery phase, it aims to find *all* potentially interesting patterns according to rather loose criteria for frequency and confidence.
2. In the presentation phase, flexible tools using *templates* are applied to iteratively and interactively create different views to the discovered patterns.

In the field of data mining or exploratory data analysis, the goal is to discover previously unknown information. That is why it can be hard, or even impossible, to specify beforehand, what is interesting. This is particularly true with telecommunication alarms, because the networks are continuously updated.

Our motivation for discovering a lot of rules once is that network management expert's requests for different viewpoints to the data can then be responded very quickly: a new pattern discovery phase is not necessary, but simply a new view to the already discovered patterns. By producing all rules at once, different views of the data can be created very efficiently in the presentation phase. The idea is that any rule that occurs frequently enough can be potentially interesting. The algorithms find all rules that fulfil the given threshold criteria, and the decision of what is interesting is for the most part left to the network manager to explore.

### 2.3. TASA structure and functions

The current version of TASA is a Java-based system with client/server architecture (see Fig. 2). The server runs on Unix-based systems (SunOS, Linux, etc.), and clients can be installed on any Java-supporting platforms.

#### 2.3.1. TASA server

The main component of the system is the TASA server. All the data sets and generated rule sets reside at the server side, and they are stored either in flat files or in a database. This way all operations that require extensive calculation power and storage capacity can be efficiently performed, and the results can be viewed by all clients. Thus, TASA clients only initiate actions at the server side and then show the results. The server is multithreaded, i.e. it can serve several clients simultaneously.

In fact, TASA server contains three kinds of servers: a file server, an RMI (Remote Method Invocation) server, and a socket server. The file server takes care of file management and process runs, while the RMI server handles and forwards remote method calls from clients. In the current system, also a socket server is needed in establishing a connection between a client and a server.

Preprocessing, i.e. preparing the raw data into a suitable form for the analysis, is done at the server side by authorized personnel using either tools provided by TASA or production database environment.

#### 2.3.2. TASA client

At the client's side, the user can browse pre-generated information about the data to be analyzed, generate association and episode rules, view generated rules and select rules from the rule collection (i.e. make views). TASA client neither needs any additional working disk space nor superior computational power.

*File Manager*. Using the File Manager window, the user can both select data sets for association and episode rule creation and generated rule sets for further analysis. The connection to TASA Server is created by selecting the login window from the main menu.

*Basic statistics*. The TASA system shows some basic statistics about the datasets, alarms of the dataset, and predicates used. The dataset statistics contains, e.g. information about the total number of alarms and their average frequency. The alarm information, in turn, lists all alarm types and, e.g. their number and percentage of all alarms.

*Association and episode rule generation*. In the rule creation part, the user can give detailed instructions for rule generation as described in Section 3.2.

*Rule selection with templates*. The rule viewing window is based on the use of templates with some additional features for defining the number of items in IF and THEN parts of the rule, and for sorting the rules. For more detailed information about the template concept, see Section 4.

## 3. Rule discovery in TASA

As already mentioned in Section 2, in TASA we consider two kinds of recurrent patterns, episode rules [10] and association rules [3]. The rule formalisms are briefly described in Section 3.1, and in Section 3.2 the rule discovery phase from the TASA user point of view is presented. Finally a simple example of the utilization of rules is described in Section 3.3.

### *3.1. Discovery methods*

In telecommunication alarm data, *alarm predicates* are the expressions used to refer to the (properties of) alarms. For episode rules, the type of the alarm and the sender of the alarm are the most typical predicates. For association rules we consider also predicates such as the priority of the alarm, the day of the week, whether the alarm occurred during office hours or not, etc.

### *3.1.1. Episode rules*

For episodes [10], the input data consists of a sequence of *events*, i.e. (*event type, time*) pairs, e.g. (*alarm 1234, 98-01-21 22:15:02*). An episode consists of the event types that tend to occur close to each other, i.e. within a given time window. Thus, if the time window is 60 s, the sequence

*(alarm 1234, 98-01-21 22:15:02)*
*(alarm 2275, 98-01-21 22:15:40)*
*(alarm 3244, 98-01-21 22:16:25)*

contains the episodes (*alarm 1234*, *alarm 2275*) and (*alarm 2275*, *alarm 3244*), but does not contain the episode (*alarm 1234*, *alarm 3244*).

In the most simple case, as above, only the alarm type is considered. The episodes reveal connections between types of alarms without respect to the network elements that sent the alarms. Alternatively, e.g. predicates specifying (sender, alarm type) pairs can be considered, making it explicit that the input is merged from alarms from several senders in the network. An episode rule found with predicates like this shows connections between different types of alarms from particular network elements. Predicates consisting of the (sender type, alarm type) pair have actually been proved to be one of the most useful forms: episode rules between types of alarms in different types of devices seem to describe the network behavior to a reasonable level of abstraction.

An episode is *parallel*, if there is no requirement for the order of the events within the time window, and *serial*, if a total order of the events is required. For example, episode (*alarm 1234*, *alarm 2275*) is parallel, if alarms 1234 and 2275 can occur in any order, whereas the episode is serial, if the alarms must occur in the given order. An episode is said to be *frequent*, if the episode occurs in the event sequence often enough, i.e. at least so many times as indicated by the *frequency threshold*. Hence, the essential parameters for defining all frequent episodes in an event sequence are the width of the time window, and the frequency threshold.

**Definition 1.** Formally, a serial episode is a sequence $\langle A_1,\ldots,A_k\rangle$ of alarm predicates. Informally, the episode corresponds to $k$ alarms that satisfy the predicates $A_i$. Given a sequence $S = \langle a_1,\ldots,a_n\rangle$ of alarms, a serial episode $\alpha = \langle A_1,\ldots,A_k\rangle$ occurs in $S$ if there is an injective mapping $f{:}\{1,\ldots,k\} \rightarrow \{1,\ldots,n\}$ such that for all $i$, $1 \le i \le k$, predicate $A_i$ is satisfied by alarm $a_{f(i)}$, and for all $i$, $1 \le i \le k - 1$ we have $f(i) < f(i + 1)$. We say that an episode $\beta = \langle B_1,\ldots,B_l\rangle$ is a subepisode of $\alpha$ if there is an injective mapping $h{:}\{1,\ldots,l\} \rightarrow \{1,\ldots,k\}$ such that for all $i$, $1 \le i \le l$ we have $A_{h(i)} = B_i$ and for all $i$, $1 \le i \le l - 1$ we have $h(i) < h(i + 1)$.

**Definition 2.** A parallel episode is a multiset $\alpha = \{A_1,\ldots,A_k\}$ of alarm predicates, and $\alpha$ occurs in given sequence $S = \langle a_1,\ldots,a_n\rangle$ if there is an injective mapping $g{:}\{1,\ldots,k\} \rightarrow \{1,\ldots,n\}$ such that for all $i$,$1 \le i \le k$, predicate $A_i$ is satisfied by alarm $a_{g(i)}$. Another unordered episode $\beta$ is a subepisode of $\alpha$ if and only if $\beta \subseteq \alpha$.

An episode rule gives the observed conditional probability that a certain combination of alarms occurs within some time bound, given that another combination of alarms has occurred within a time bound.

In Table 1 the episode rule format as used in TASA is described. In this format, the IF part refers to the left-hand side of the rule, and the THEN part to the right-hand side of the rule, respectively. The WITH part contains additional

Table 1
Episode rule format used in TASA

| Episode rules | |
|---|---|
| IF | ⟨alarm 1⟩ |
| | … |
| | ⟨alarm n⟩ |
| THEN | ⟨alarm 1⟩ |
| | … |
| | ⟨alarm n⟩ |
| WITH | [w1] [w2] *C* (*F*1/F2) |
| | … |
| | [w(*n*-1)] [w*n*] C (F1/F2) |
| w1…w(*n* − 1) | = time bound for IF side |
| w2…w*n* | = time bound for whole rule |
| *C* | = rule confidence |
| *F*1 | = whole rule frequency |
| *F*2 | = IF side frequency |

information, e.g. about the rule strength and frequency. For example, we can have rule

IF link alarm
link failure
THEN high fault rate
WITH [20] [40] conf(0.23) freq(246/1056)

which tells us that in 23% of cases, where *link alarm* and *link failure* occurred within 20 s, *high fault rate* occurred within 40 s. Moreover, in the data the left-hand side occurred 1056 times and in 246 cases it was followed by the right-hand side, within the given time windows.

**Definition 3.** Formally, an episode rule is an expression $\beta$[win1] $\Rightarrow$ $\alpha$[win2], where $\beta$ and $\alpha$ are episodes such that $\beta$ is a subepisode of $\alpha$, and $\text{win}_1$ and $\text{win}_2$ are integers. The interpretation of the rule is that if episode $\beta$ has a minimal occurrence at $[t_s,t_e]$ with $t_e - t_s \le win_1$, then the whole superepisode $\alpha$ occurs at interval $[t_s, t'_e]$ for some $t'_e$ such that $t'_e - t_s \le win_2$.

### 3.1.2. Association rules

To capture relationships between alarm predicates, we use *association rules* and *frequent sets*. An association rule is an expression $X \Rightarrow Y$, where $X$ and $Y$ are sets of predicates. Given a set of alarms, the *confidence* of such rule is the observed conditional probability with which predicates in $Y$ are satisfied by an alarm given that predicates in $X$ are. The rule is called *frequent*, if its *frequency* exceeds a user-given threshold; i.e. if all the predicates in $X \cup Y$ occur together at least a user-specified minimum number of times.

Table 2
Association rule format used in TASA

| Association rules | |
|---|---|
| IF | ⟨attribute 1⟩ |
| | … |
| | ⟨attribute n⟩ |
| THEN | ⟨attribute 1⟩ |
| | … |
| | ⟨attribute n⟩ |
| WITH | conf(*C*) freq (*F*) |
| *C* | = rule confidence |
| *F* | = rule frequency |

In Table 2 the association rule format as used in TASA is described. For instance, the rule

IF sender = EL1
THEN *alarm_type* = 1234
WITH conf(0.70) freq(0.12)

states that 70% of the alarms that are sent by network element ‘‘EL1’’ are of type ‘‘1234’’, and that this is true for 12% of all the alarms in the analyzed data set.

### 3.2. Rule discovery

In TASA, the user can create and query episode and association rules. In Fig. 3 there is an example of what information can be given to the system in order to create episode rules.

The user can specify, e.g. the following parameters for the discovery of both association and episode rules.

- Rule type: unordered/ordered (required).
- Frequency threshold (required).
- Confidence threshold (required).
- Set of time bounds or maximum time bound (for episode rules; required).
- Maximum rule size.
- Bounds for the sizes of both rule left-hand and right-hand size.

For both episode and association rules, a *frequency threshold* is given by the user. The method outputs all episode and association rules specified by the parameters above, *whose frequency is at least the user-specified threshold*.

The method is aimed at discovering statistical rules, not spotting interesting individual cases. With the frequency threshold the user is able to filter out rules that are too rare to be trustworthy. For instance, with a frequency threshold of 20 an episode is output, only if it appears at least 20 times in the analyzed database. The algorithm is complete in this respect: it is guaranteed to output all episodes that have at least 20 occurrences.

The frequency threshold is crucial for the running time of the algorithm. If the threshold is low, then rules that occur rarely are included in the output, and the discovery time is longer. Suitable values depend heavily on the nature and amount of data. For a database with 100 000 alarms, thresholds in the range of 50–500 may be reasonable.

For both episode and association rules, the user also specifies a *confidence threshold c*. The algorithm then outputs all episode and association rules *whose confidence is at least c*.

The confidence threshold allows the user to filter out rules that are too weak to be useful. For instance, a confidence threshold of 80% limits the output to rules that hold with at least 80% certainty. The confidence threshold has no particular effect on the running time, so it is useful to specify a low threshold in the rule discovery phase and to prune weak rules later interactively with the user interface tools.

For episode rules, the user also supplies a set *W* of time bounds with which the rules are constructed. Two aspects guide the setting of *W*.

1. The maximum time bound in *W* should be larger than the maximal temporal duration of the phenomena that are searched for.
2. The number of time bounds in *W* directly affects the temporal granularity at which episode rules are found.

Fault management experts have typically preferred time bounds ranging from 5 s to 10 min, e.g. with roughly logarithmically growing time bounds 5, 10, 30 s, and 1, 2, 5, 10 min. The higher is the number of time bounds, the higher is the number of rules also. The effect on the running time is not strong.

The discovery method outputs each episode rule and association rule satisfying the above conditions. The conditions should typically be quite loose, so large amounts of rules are discovered. For each rule, TASA outputs the confidence and the frequency.

In the pattern discovery phase, the parameters have to be adjusted properly. For example, in the case of telecommunication network alarm data there usually is a roughly known time window size within which interesting relationships exist. If the window size is not set properly, relationships can remain hidden if they do not fit into the window, or relationships can be buried under noise, if the window is too large.

### 3.3. Knowledge utilization

We use as an example an alarm correlation system which operates in real time and is also able to handle delayed alarms and slightly inaccurate time stamps. In the correlation patterns, delays are handled with a special *wait* function. Episode and association rules can be applied in this system in a rather straightforward way. For instance, the rule

```
IF      link alarm
        link failure
THEN    high fault rate
WITH    [5]   [60]    conf(0.7)    freq(740/1056)
```

discovered by TASA can be coded in the system as follows:

```
If ‘‘alarm type = link alarm’’ then
   start time;
   wait until ‘‘alarm type = link failure’’ or ‘‘time = 5 s’’;
   if ‘‘alarm type = link failure’’ then
      send an alarm ‘‘high fault rate with 70% probability
      in 60 s’’
   else forward the original alarm;
   fi;
fi
```

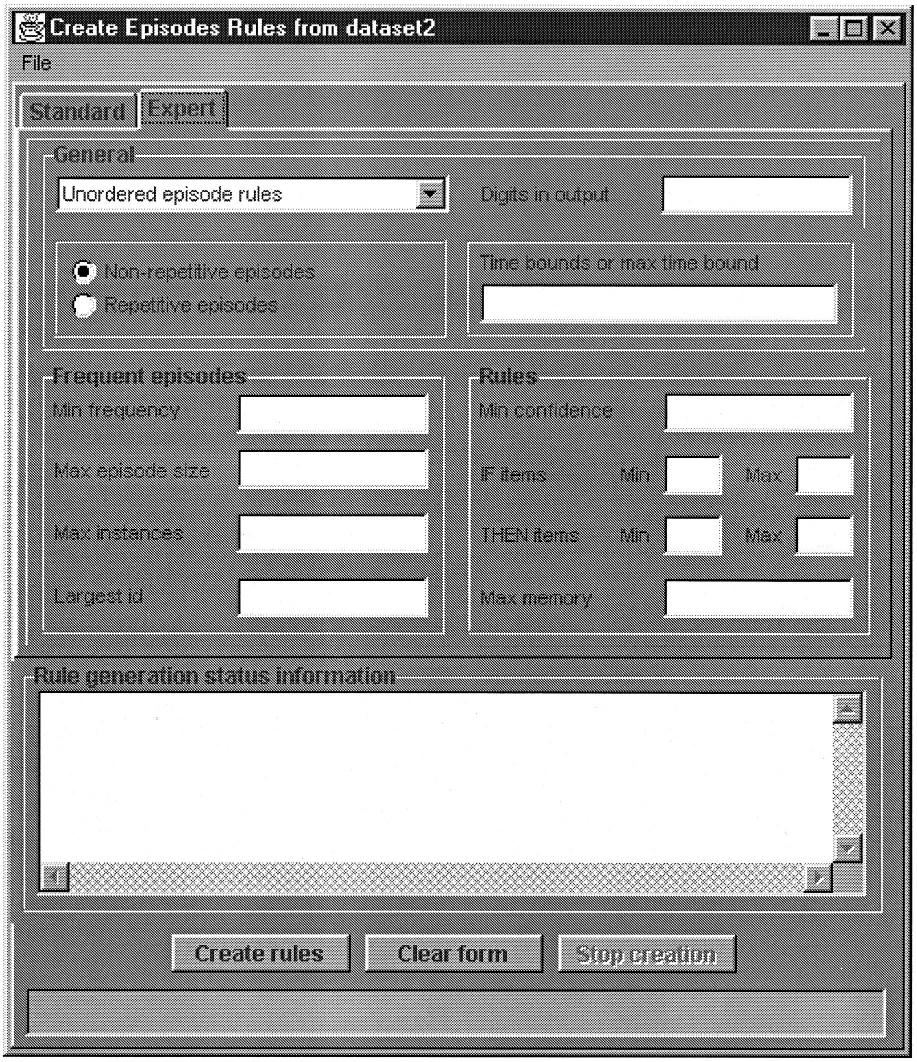
That is if a link alarm occurs and a link failure follows within 5 s, the rule right-hand side information is sent and the original alarms are suppressed. If a link failure does not follow within 5 s, the original link alarm is forwarded.

## 4. Iterative search with templates

The presentation of discovered knowledge is a main part of our methodology. In this phase the interesting patterns should be located from large collections of potentially interesting patterns. But what is interesting? How to define the interestingness? As the goal of knowledge discovery is to find useful patterns, an obvious requirement for success in this task is that the knowledge representation formalisms are such that the user is able to find and understand the useful patterns. In this section, we present methods for exploring large sets of association and episode rules. The ideas of this section can be best applied on large unstructured sets of rules and other similar, simple pattern formalisms.

The datasets in Table 3 represent typical usage of TASA system in analyzing both short-term sequences, a couple of days, and long-term sequences, a couple of months. They also reflect the real-world situation, where the material to be analyses contains plenty of different types of events, i.e. hundreds or thousands of types of alarms. In the resulting rules, however, many alarm types are not present at all due to their low frequencies that do not exceed the given thresholds. The experiences have indicated that the algorithms are not well suited for analyzing event sequences that contain long bursty periods. It is sometimes more useful to cut off and analyze such periods separately.

We have evaluated the efficiency of the episode discovery algorithm using several alarm databases from fixed and cellular networks. Typical running times on a Pentium-based PC range from few seconds to an hour, depending on the database and the parameters. Episodes with the alarm type as the only predicate can be discovered in a sequence of about 100 000 alarms with a window width of 60 s from few seconds to some tens of seconds. The time requirement increases slowly as more time bounds are used, but the time increases more slowly than the number of rules. The time requirement of the algorithm is linear in the number of alarms, and much larger databases can be analyzed with acceptable response times. In the case of association rules with possibly thousands of different predicates the discovery can take an hour or even more.



Unordered episode rules

● Non-repetitive episodes
○ Repetitive episodes

Fig. 3. Episode rule creation in TASA with expert mode; standard mode asks only for the required parameters.

That is because the predicates are derived by considering about every bit of information contained in alarm messages, e.g. ‘‘seconds = 0’’, ‘‘seconds = 1’’, etc.

The number of frequent rules decreases rapidly as the frequency threshold increases. Also, although the initial number of rules may be quite large, it decreases fairly rapidly if we require a reasonable confidence. As can be seen in Table 3, the method can easily produce very large amounts of rules-even when the threshold values have been properly selected. In fact, many of the rules discovered by TASA are deemed trivial or uninteresting by the network managers:

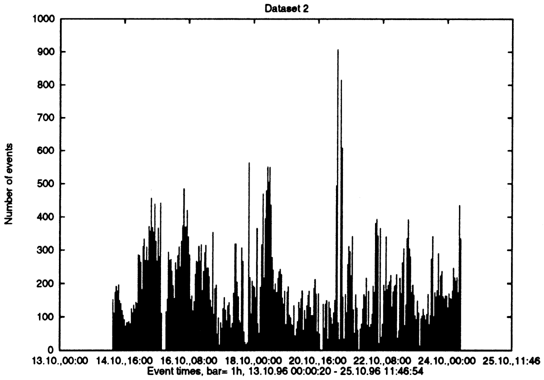
- *A rule can correspond to prior knowledge or expectations*. For instance, we might know from the network implementation that if an element sends an alarm *A*, it will also send an explanatory alarm *B*.
- *A rule can refer to uninteresting attributes or attribute combinations*. In the case of the network alarm data, rules containing, e.g. low severity alarms may be non-interesting, and the user would like to filter out all such rules except for some special situations.
- *Rules can be redundant*. In the alarm data the rules can contain alarms of different abstraction levels but actually referring to the same fault.

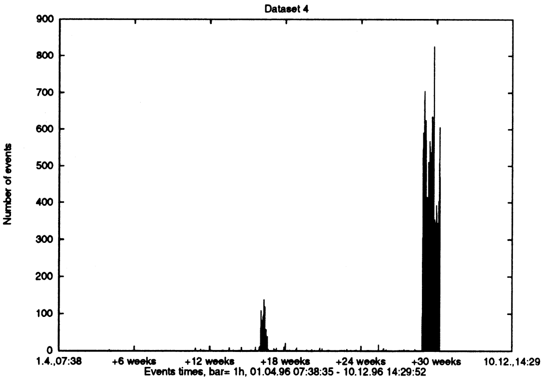
For the most part, what is interesting depends on the case, and is highly based on the user’s personal aims and perspective (see discussion, e.g. in [4,13]). The knowledge trivial to one expert may not be trivial to another, but with proper tools each expert may filter the rule collection based on his/her personal background knowledge.

TASA offers a variety of focusing and ordering criteria for rules and supports iterative retrieval from the discovered knowledge. Network management experts can manipulate the rule set using selection and sorting operations, as well as more complex operations for including or excluding certain classes of rules.

While creating a focus, simple threshold-like restrictions, such as rule frequency and confidence may satisfy a large

Table 3
Example dataset characteristics and alarm occurrence times





| | | | |
|---|---|---|---|
| First alarm | 13.10.96 at 00:00:20 | First alarm | 01.04.96 at 07:38:35 |
| Last alarm | 25.10.96 at 11:46:54 | Last alarm | 10.12.96 at 14:29:52 |
| Duration | 12d:11h:46m:34s | Duration | 253d:6h:51m:17s |
| Number of alarms | 58 616 | Number of alarms | 98 413 |
| Number of alarm types | 1 956 | Number of alarm types | 212 |
| Frequency threshold | 25 | Frequency threshold | 50 |
| Confidence threshold | 0.2 | Confidence threshold | 0.2 |
| Window sizes | 5,10,20,30,60 | Window sizes | 5,10,20,30,60 |
| Number of distinct rules | 778 | Number of distinct rules | 9 406 |

number of rules. In TASA, this problem can be alleviated by selecting rules to or removing rules from the view by *templates* [7]. Although being rather simple, this technique is surprisingly powerful.

**Definition 4.** We define templates as simple regular expressions that describe, in terms of alarm predicates, the form of rules that are to be shown or not shown. More formally, a template is an expression

$$A_1,\ldots,A_k \Rightarrow A_{k+1},\ldots,A_l,$$

where each $A_i$ is either an alarm predicate, the name of an alarm predicate collection, or an expression $C+$ or $C*$, where $C$ is a collection name.[3] Here $C+$ and $C*$ correspond to one or more and zero or more instances of the collection $C$, respectively. A rule

$$B_1,\ldots,B_m \Rightarrow B_{m+1},\ldots,B_n$$

matches a template if the rule can be considered to be an instance of the pattern.

In TASA, the template concept is implemented as in Fig. 4. In the case of episode rules, the user can define the alarms that can occur in the rule, and also the bounds for, e.g. frequency, confidence, and number of alarms in one rule. Some scenarios utilizing the iteration-based analysis of a rule base are sketched in Examples 5 and 6.

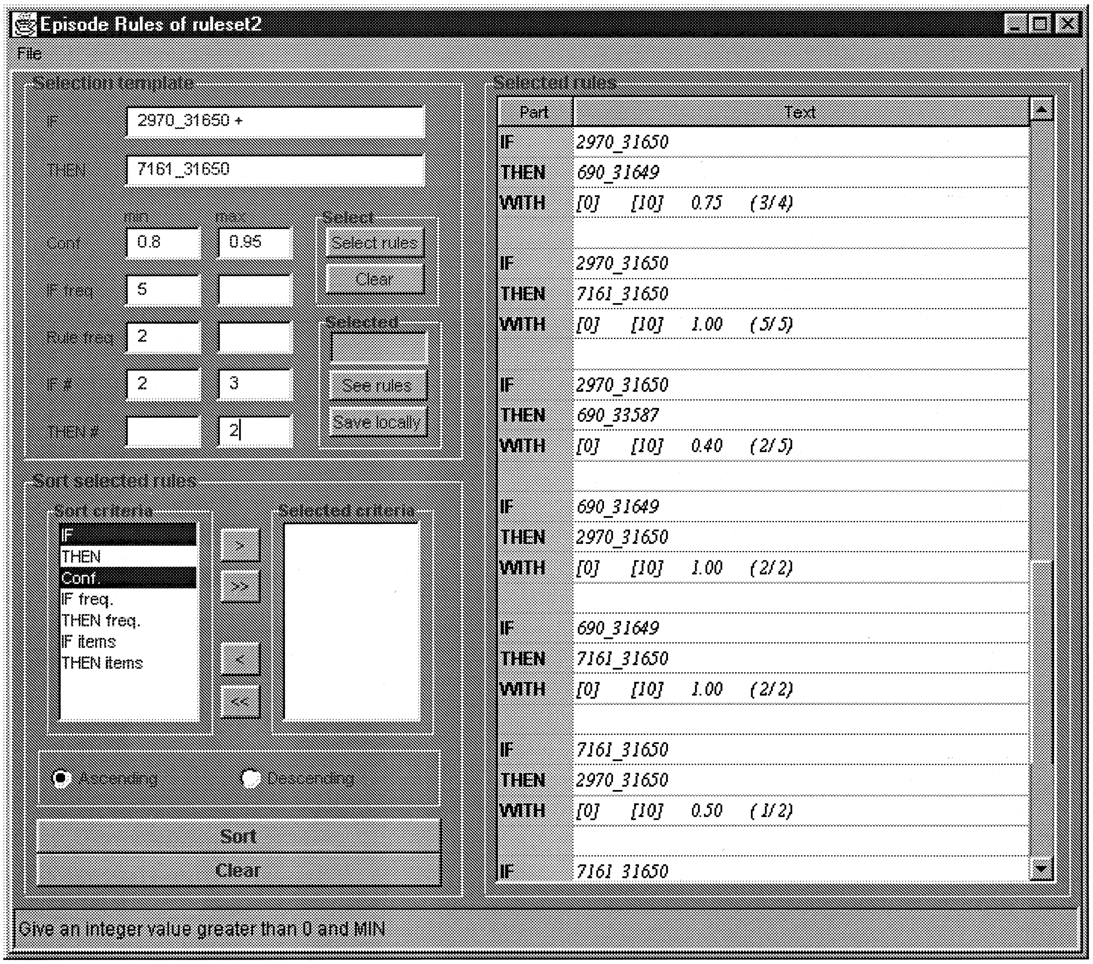
**Example 5.** Focus can be set to, e.g. day-time alarms by selecting only association rules that contain the predicate ‘‘office hours = yes’’. Or, episode rules containing alarms from separate subnetworks can be obtained by using templates that reject all rules where the senders are in the same subnetwork.

The template concept can be combined with thresholds for rule frequency, confidence, and significance. The user may state restrictions such as ‘‘rule frequency must be between 5 and 30%’’, ‘‘rule confidence must be at least 80%’’, and ‘‘rule significance must be over 0.95’’. In this case the user filters out very rare and reasonably frequent rules, and further on selects only those which are both strong and statistically significant.

Several *positive* and *negative* templates can be used simultaneously to achieve the desired viewpoint. To be shown, a rule must match all positive templates and none of the negative ones.

**Example 6.** As an example of how the system can be used

[3] $A_i$ can also be a regular expression.



Selection template

|  | min | max |
|---|---|---|
| IF | 2970_31650 + |  |
| THEN | 7161_31650 |  |
| Conf | 0.8 | 0.95 |
| IF freq | 5 |  |
| Rule freq | 2 |  |
| IF # | 2 | 3 |
| THEN # |  | 2 |

Select

Selected

Sort selected rules

Sort criteria: IF, THEN, Conf., IF freq., THEN freq., IF items, THEN items

Selected criteria

● Ascending ○ Descending

Selected rules

| Part | Text |
|---|---|
| IF | 2970_31650 |
| THEN | 690_31649 |
| WITH | [0] [10] 0.75 (3/4) |
|  |  |
| IF | 2970_31650 |
| THEN | 7161_31650 |
| WITH | [0] [10] 1.00 (5/5) |
|  |  |
| IF | 2970_31650 |
| THEN | 690_33587 |
| WITH | [0] [10] 0.40 (2/5) |
|  |  |
| IF | 690_31649 |
| THEN | 2970_31650 |
| WITH | [0] [10] 1.00 (2/2) |
|  |  |
| IF | 690_31649 |
| THEN | 7161_31650 |
| WITH | [0] [10] 1.00 (2/2) |
|  |  |
| IF | 7161_31650 |
| THEN | 2970_31650 |
| WITH | [0] [10] 0.50 (1/2) |
|  |  |
| IF | 7161 31650 |

Give an integer value greater than 0 and MIN

Fig. 4. Rule Viewing window of the new TASA system containing a template skeleton with additional functionalities.

for off-line network surveillance, consider the following typical scenario. Assume the network manager has used TASA to discover association rules for the current month. First he might want to see what the alarms have been like during the current week, say week 30, so he uses a template to select rules with the predicate ‘‘week = 30’’ as the left-hand side.

The number of selected rules is still very large. The network manager decides to restrict the rule right-hand side to only contain one predicate, and he also sorts the rules by their confidences.

Looking at the selected rules, he sees the rule ‘‘if week = 30 then alarm type = connection failure’’ with confidence 0.12, and he infers that an unusually large fraction of alarms during the week has been of type connection failure. To see in more detail what the alarms have been like, he refines the template and selects rules with ‘‘week = 30 and alarm type = connection failure’’ as the left-hand side.

Looking at the new set of selected rules, the network manager sees that a lot of rules concern the network element EL1. That reminds him of maintenance undertaken in the beginning of the week that explains those rules. To remove the rules, he applies a negative template with the predicate ‘‘network element = EL1’’.

The resulting set of rules shows nothing special, but just to make sure the network manager wants to compare the rules with the corresponding rules from some previous week. He opens a copy of the window, and changes the first template to ‘‘week = 29’’. If there is anything special or interesting, the viewing criteria can be refined or altered again.

## 5. Related work

There are several KDD formalisms that fit in the methodological setting used in TASA. Association rule [3] and episode [2] algorithms can efficiently discover thousands of rules from relatively simple databases (see, e.g. Refs. [10,14–16] for efficient algorithms and [7] for the large number of rules). A formal treatment of the setting of discovering all interesting sentences and an analysis of a general algorithm can be found in [17].

In TASA, we use association and episode rules, but the basic idea—iteration in the pattern presentation phase—can be applied to formalisms that have some similar properties as association and episode rules:

- There is an algorithm that produces lots of potentially interesting patterns.
- The time requirement for discovering all potentially interesting patterns is not considerably greater than if the discovery was focused to a small subset of the potentially interesting patterns.
- The desired focus is not known definitely in advance.

This approach, which allows the user to set the focus after the pattern discovery phase, and where the user has a total, explicit control over the resulting rule set, is similar to the one used in some ILP (Inductive Logic Programming) systems, e.g. Claudien [18]. In Claudien, however, the focus is set before the discovery, and the user has control over result by the use of language bias in ILP.

Hoschka and Klösgen [19] have also used templates for defining interesting knowledge, and their ideas have strongly influenced our work. Their approach is based on few fixed statement types and partial ordering of attributes, whereas our approach is closer to regular expressions.

This approach of discovering all patterns can be contrasted with numerous methods, e.g. in machine learning, which are more focused and produce one or at most a few patterns that match the given problem specification. These methods usually require that the searched or learned subject is quite carefully described in advance, and they leave any other potentially interesting phenomena hidden. The advantage of these systems is that the patterns they find are more expressive than the relatively simple association and episode rules, and focusing the pattern discovery is thus more important.

For instance, Explora [13,19] finds interesting instances of statistical patterns. In Explora the pattern discovery phase is focused by the user. The system selects and presents the best patterns to the user, and, based on the results, the user can change the focus and repeat the pattern discovery. The patterns discovered by 49er [20] are contingency tables, equations, and logical equivalence. The user can interactively change the focus, e.g. independent and dependent variables, and require for a new pattern discovery. The Key Finding Reporter (Kefir) [21,22] discovers and explains deviations, and gives recommendations for corrective actions. Applications of Kefir are tailored with a lot of domain knowledge to be aware of the interestingness criteria, corrective actions, etc. of the domain. Given a database from the domain, a Kefir-based application produces a report of the deviations without iteration.

## 6. Conclusions and experiences

Different versions of TASA have been in prototype use in four telecommunication companies since the beginning of 1995. TASA has been found useful in, e.g. finding long-term, rather frequently occurring dependencies, creating an overview of a short-term alarm sequence, and evaluating the alarm data base consistency and correctness.

Unexpected dependencies have been found, e.g. between network elements which are not closely connected in the network topology. An example of such a dependency is that when a remote device sends alarms, the fault is reflected to another corner of the network through several devices, and not always necessarily via the same routes and devices. So, just analyzing the neighboring devices might not reveal any strong relationships. However, when a larger region is analyzed, such a relationship can be detected. Beginning from the first tests, discovered rules have been integrated into alarm correlation systems.

However, many of the rules discovered by TASA are deemed trivial by the network managers. Some of the rules correspond to the knowledge that the network managers have about the behavior of the network, and some other rules reflect the assumed functioning of network devices. Luckily, much of the trivial knowledge can be expressed and removed with templates. Templates are also useful since the knowledge trivial to one expert may not be trivial to another, and with templates each expert may filter the rule collection based on his/her personal background knowledge.

The usability of discovery tools has an essential, often perhaps under-estimated role. The usability of an early version of TASA was tested in the usability laboratory of the Helsinki University of Technology. The tests contained, e.g. user tests taken by four fault management experts from telecommunication companies. In the tests, TASA was generally acknowledged as appealing. However, first-time users were unfamiliar with many concepts from the knowledge discovery field. Despite these problems with the terminology, the system as a whole got encouraging comments.

Overall, TASA has been considered useful. Episode rules are being used as first drafts of correlation rules, whereas association rules are more typically used for creating short-term overviews in off-line network surveillance. Telecommunication operators are integrating these methods to their alarm analysis and surveillance systems.

## Acknowledgement

This research has been supported by the Technology Development Center of Finland (TEKES) and the Academy of Finland.

## References

[1] K. Hätönen, M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, Knowledge discovery from telecommunication network alarm databases, Proceedings of the 12th International Conference on Data Engineering (ICDE'96), New Orleans, Louisiana, IEEE Computer Society Press, Silver Spring, MD, 1996 pp. 115–122.

[2] H. Mannila, H. Toivonen, A.I. Verkarno, Discovering frequent

episodes in sequences, Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD'95) Montreal, Canada, AAAI Press, Menlo Park, CA, 1995 pp. 210–215.

[3] R. Agrawal, T. Imielinski, A. Swami, Mining association rules between sets of items in large databases, in: P. Buneman, S. Jajodia (Eds.), Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'93), Washington DC, USA, ACM, New York, 1993.

[4] G. Piatetsky-Shapiro, Discovery, analysis, and presentation of strong rules, in: G. Piatetsky-Shapiro, W.J. Frawley (Eds.), Knowledge Discovery in Databases, AAAI Press, Menlo Park, CA, 1991, pp. 229.

[5] A. Silberschatz, A. Tuzhilin, What makes patterns interesting in knowledge discovery systems, IEEE Transactions on Knowledge and Data Engineering 8 (6) (1996) 970–974.

[6] W. Kloesgen, Explora: a multipattern and multistrategy discovery assistant, in: U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, CA, 1996, pp. 249.

[7] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, A.I. Verkamo, Finding interesting rules from large sets of discovered association rules, Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94), Gaithersburg, MD, ACM, New York, 1994.

[8] M. Klemettinen, H. Mannila, H. Toivonen, Rule discovery in telecommunication alarm data, Journal of Network and Systems Management, accepted for publication.

[9] G. Jakobson, M.D. Weissman, Alarm correlation, IEEE Network 7 (6) (1993) 52–59.

[10] H. Mannila, H. Toivonen, A.I. Verkamo, Discovery of frequent episodes in event sequences, Data Mining and Knowledge Discovery 1 (3) (1997) 259–289.

[11] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, From data mining to knowledge discovery: an overview, in: U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, CA, 1996, pp. 1.

[12] K. Hätönen, M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, Telecommunication alarm sequence analyzer, or how to enjoy faults in your network, Proceedings of the IEEE Network Operations and Management Symposium (NOMS'96), 1996, pp. 520–529.

[13] W. Kloesgen, Efficient discovery of interesting statements in databases, Journal of Intelligent Information Systems 4 (1) (1995) 53–69.

[14] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, A.I. Verkamo, Fast discovery of association rules, in: U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, CA, 1996, pp. 307.

[15] A. Savasere, E. Omiecinski, S. Navathe, An efficient algorithm for mining association rules in large databases, Proceedings of the 21st International Conference on Very Large Data Bases (VLDB'95), Zürich, Switzerland, 1995, pp. 432–444.

[16] H. Toivonen, Sampling large databases for association rules, Proceedings of the 22nd International Conference on Very Large Data Bases (VLDB'96), Mumbai, India, Morgan Kaufmann, Los Altos, CA, 1996 pp. 134–145.

[17] H. Mannila, H. Toivonen, Levelwise search and borders of theories in knowledge discovery, Data Mining and Knowledge Discovery 1 (3) (1997) 241–258.

[18] L. De Raedt, L. Dehaspe, Clausal discovery, Machine Learning 26 (2) (1997) 99–146.

[19] P. Hoschka, W. Klösgen, A support system for interpreting statistical data, in: G. Piatetsky-Shapiro, W.J. Frawley (Eds.), Knowledge Discovery in Databases, AAAI Press, Menlo Park, CA, 1991, pp. 325.

[20] R. Zembowicz, J.M. Zytkow, From contingency tables to various forms of knowledge in databases, in: U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, CA, 1996, pp. 329.

[21] C.J. Matheus, G. Piatetsky-Shapiro, D. McNeill, Selecting and reporting what is interesting, in: U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, CA, 1996, pp. 495.

[22] G. Piatetsky-Shapiro, C.J. Matheus, The interestingness of deviations, in: U.M. Fayyad, R. Uthurusamy (Eds.), Knowledge Discovery in Databses, Papers from the 1994 AAAI Workshop (KKD'94), Seattle, Washington DC, 1994, pp. 25–36.