

Mining Sequential Alarm Patterns in a Telecommunication Database

Pei-Hsin Wu, Wen-Chih Peng and Ming-Syan Chen

Department of Electrical Engineering,
National Taiwan University
Taipei, Taiwan, ROC
{mschen@cc.ee.ntu.edu.tw, wcpeng, peggywu@arbor.ee.ntu.edu.tw}

Abstract. A telecommunication system produces daily a large amount of alarm data which contains hidden valuable information about the system behavior. The knowledge discovered from alarm data can be used in finding problems in networks and possibly in predicting severe faults. In this paper, we devise a solution procedure for mining sequential alarm patterns from the alarm data of a GSM system. First, by observing the features of the alarm data, we develop operations for data cleaning. Then, we transform the alarm data into a set of alarm sequences. Note that the consecutive alarm events exist in the alarm sequences, and it is complicated to count the occurrence counts of events and extract patterns. Hence, we devise a new procedure to determine the occurrence count of the sequential alarm patterns in accordance with the nature of alarms. By utilizing time constraints to restrict the time difference between two alarm events, we devise a mining algorithm to discover useful sequential alarm patterns. The proposed mining algorithm is implemented and applied to test against a set of real alarm data provided by a cellular phone company. The quality of knowledge discovered is evaluated. The experimental results show that the proposed operations of data cleaning are able to improve the execution of our mining algorithm significantly and the knowledge obtained from the alarm data is very useful from the perspective of network operators for alarm prediction and alarm control.

1 Introduction

Due to recent technology advances, an increasing number of users are accessing various information system via wireless communication. Such information systems as stock trading, banking, wireless conferencing, are being provided by information services and application providers [5][6][7][13], and mobile users are able to access such information via wireless communication from anywhere at anytime [3][12][15].

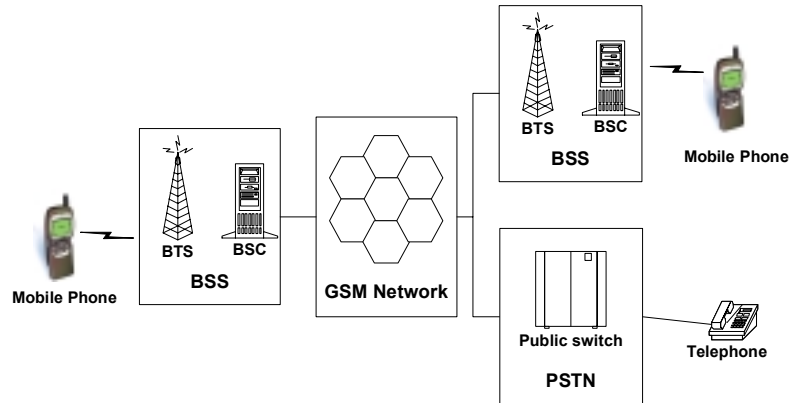


Fig. 1. The general architecture of a GSM system.

The architecture of wireless networks varies from one standard to another. Use of the *Global System for Mobile Communication (GSM)* continues to spread throughout the world. Figure 1 shows the general architecture of the GSM system. It can be seen that a GSM system comprises lots of interconnected components and each component also contains several subcomponents. Individual components and subcomponents generate alarms indicating some sort of abnormal situations. Table 1 shows an example of selected real alarm data where sourceID is the identification of an individual component that generates the alarm and errorID indicates the abnormal situation. Thus, a GSM system produces daily a large amount of alarm data which contains hidden valuable information about the system behavior. The knowledge discovered from alarm data can be used in finding problems in networks and possibly in predicting severe faults. Since the number of alarms could be in tens of thousands daily, it is infeasible to investigate these alarms manually. In order to extract the valuable knowledge from the alarm data, we utilize the techniques of data mining in this study.

Table 1. An example of selected alarm data.

day	time	Source	sourceID	errorID	...
2001/01/01	15:07:09	SITE	21233	a	...
2001/01/01	15:46:23	SITE	21233	f	...
2001/01/01	19:04:37	BSC	bsc05	z	...
2001/01/01	22:15:44	SITE	33009	h	...
2001/01/01	23:20:15	SITE	10021	q	...
...

Data mining, which is also referred to as *knowledge discovery* in databases, means a process of extracting implicit, previously unknown and potentially useful information (in terms of knowledge rules, constraints, regularities) from data in databases. By knowledge discovery in databases, interesting knowledge, regularities, or high-level information can be extracted from the relevant sets of data in databases

and be investigated from different angles. Various data mining capabilities have been explored in the literature [1][2][4][8][9][10][16]. It is noted that utilizing the technique of mining sequential patterns is able to extract valuable knowledge from the alarm data generated by a GSM system [11][14]. In mining sequential patterns, the input data is a set of sequences, called *data-sequences*. Each data-sequence is a list of transactions, where each transaction is a set of literals, called *items*. Typically there is a transaction-time associated with each transaction. A *sequential pattern* also consists of a list of sets of items. The problem is to find all sequential patterns with a user-specified minimum *support*, where the support of a sequential pattern is the percentage of data-sequences that contain the pattern. In prior works, the constraint imposed by the time difference between two consecutive events has not been explored. For example, the customers typically rent "Star Wars" first, and then "Empire Strikes Back". Note that such a pattern did not contain the time difference of these two renting activities. Hence, the knowledge of time difference has not been taken into account when the mining results are produced. However, such a time difference between two events is in fact an important knowledge when mining the alarm data since, with this knowledge, one can not only judge the relevance of these two events, but also predict the alarm sequence and take proper steps to prevent the occurrence of the alarms if all possible.

An example sequential alarm sequence that we shall explore in the alarm data is shown in Figure 2. The number in each circle represents the errorID, same as in Table 1, and $T_{i,j}$ denotes the time difference between alarm event_{*i*} and alarm event_{*j*}. As mentioned above, this knowledge is important from the perspective of network operators in that a network operator can hence be warned beforehand of severe alarms and take proper provision. For example, if the network operator detects that the alarm a occurring at time t , he/she should dissipate this alarm before the time $t + T_{a,q}$ to alleviate the abnormal situations incurred. It can be seen that the problem of mining sequential patterns for the telecommunications is intrinsically different from that for the rental industry, and particularly ought to capture the presence of time difference between events. Therefore, it is necessary to develop a new mining algorithm for mining constrained sequential alarm patterns for a telecommunication system.

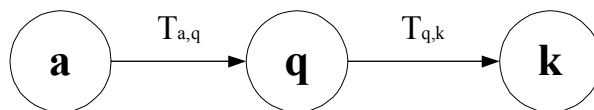


Fig. 2. An example of sequential alarm patterns.

In this paper, we devise a solution procedure for mining sequential alarm patterns from the alarm data of a GSM system. First, by observing the features of the alarm data, we develop two operations for data cleaning. Specifically, one operation is to delete those undetermined alarm events and the other is to merge the repeated alarm events caused by the same abnormal situation. After the data cleaning procedure, we transform the alarm data into a set of alarm sequences. Note that the consecutive alarm events exist in the alarm sequences, and it is complicated to count the occurrence counts of events and extract meaningful alarm patterns. Hence, we devise

a new counting method to determine the occurrence count of the sequential alarm patterns in accordance with the nature of alarms. By utilizing time constraints to restrict the time difference between two alarm events, we devise an algorithm of mining sequential alarm patterns (to be referred to as algorithm MSAP) to discover useful sequential alarm patterns. The algorithm MSAP is implemented and applied to test against a set of real alarm data provided by a cellular phone company in Taiwan. The quality of knowledge discovered is evaluated. The experimental results show that the proposed operations of data cleaning are able to improve the execution of our mining algorithm significantly and the knowledge obtained from the alarm data is very important from the perspective of network operators for alarm prediction and alarm control.

This paper is organized as follows. Preliminaries are given in Section 2. In Section 3, we devise the constraint-based algorithm for mining sequential alarm patterns. Experimental results are presented in Section 4. This paper concludes with Section 5.

2 Preliminaries

To facilitate the presentation of this paper, some preliminaries are given in this section. In Section 2.1, we describe the alarm data in a GSM system. In Section 2.2, the procedure of mining sequential alarm patterns is presented.

2.1 Alarm Data in GSM

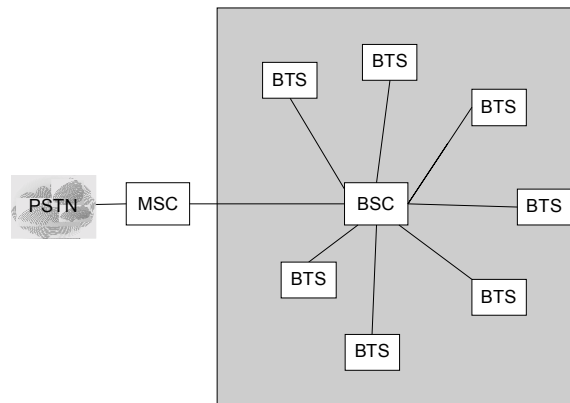


Fig. 3. The components of a GSM system: MSC, BSC, and BTS.

Figure 3 shows the components of a GSM system. The Mobile Services Switching Center (MSC) is a complete exchange system which is able to route calls from the fixed network to an individual mobile station. The MSC has connections to other entities of the GSM system. Those connections allow the MSC to gather the current statuses of mobile stations. The Base Station Controller (BSC) provides the management of the Base Transceiver Station (BTS) and also the communication

between the MSC and the BTS. The BTS supports the radio transmission/reception and the control of radio functions.

When detecting an anomalous situation, the GSM system generates an alarm event to alert the operators for this anomalous situation. All these alarm events are recorded in the alarm log table, which has several attributes, such as day, time, source, sourceID, severity and errorID. The attributes *day* and *time* record the time when the alarm event occurred. The attribute *source* is the type of the component generating the alarm event. There are three types of sources, cell, site, and BSC. Explicitly, a cell refers to a cell inside a BTS, a site refers to a base station and BSC is the base station controller. As to the other attributes of alarm log table, *sourceID* is the identification of an individual component and *severity* means the severity of the anomalous situation. Each anomalous situation or alarm event is associated with the identification number, denoted as *errorID*. For example, errorID "m" indicates that "unavailable radio signaling channels threshold critical" occurs.

2.2 Procedure of Mining Alarm Sequential Patterns

As described before, a GSM system produces daily large amount of alarm data in the alarm log data. In this paper, we explore the problem of mining sequential alarm patterns from the alarm log data. The procedure of mining sequential alarm patterns that involves the following tasks:

1. Data cleaning: In order to extract useful sequential alarm patterns, data cleaning and data preprocessing are needed [10]. The operations of data cleaning include selecting those attributes needed for data mining and removing those redundant information or outliers.

2. Discovery of patterns: After the process of data cleaning, we employ a data mining algorithm to discover the valuable sequential alarm patterns from the alarm data. The sequential alarm patterns contain not only the temporal relationship but also the time spans among the alarm events.

With the problem and the corresponding tasks described above, we then develop an approach to mining sequential alarm patterns.

3 Mining Sequential Alarm Patterns

In this section, we describe the process of data cleaning in Section 3.1. In Section 3.2, we describe the algorithm of mining sequential alarm patterns (MSAP) to extract useful sequential alarm patterns.

3.1 Data Cleaning

The goal of data cleaning is to identify the available data sources and extract the data that is needed for preliminary analysis in preparation for future mining. Clearly, the process of data cleaning depends on the patterns to be mined. Note that the alarm log data contains not only useful information but also dummy information. Before mining

sequential alarm patterns, the procedure of data cleaning is needed. Specifically, we have the following two operations for data cleaning:

Cleaning undetermined alarm events

Some anomalous causes that cannot be recognized by the alarm system are assigned their errorID to be 88888 and the alarm event with its errorID 88888 is frequently found in the alarm log data. Because of the frequent occurrence of the alarm event 88888, the sequential alarm patterns are those patterns containing lots of the alarm event 88888. Clearly, these sequential alarm patterns are of less interest. Thus, we shall clean undetermined alarm events to improve the quality of the sequential alarm patterns mined.

Merging repeated alarm events

It is also noted that some repeated alarms appear frequently in the alarm log data. It can be verified that some repeated alarms are independent events and the others are caused by the same abnormal situations. For example, the alarm log data has the sequence <AAAABC> in which alarm event A repeats four times. Notice that if these repeated alarm events are from the same abnormal situation, we would like to merge them to one alarm event (i.e., the <AAAABC> becomes <ABC>). Otherwise, we retain the repeated alarms. Note that not all repeated alarm events could be merged. Merging the repeated alarm events should be done in accordance with the *ceased table* in the telecommunication database system. The ceased table contains the time when an alarm event is generated and the time when the alarm event is ceased. Table 2 shows an example of a ceased table, where Dtime is the generated time of an alarm, Ctime is the ceased time of this alarm and Duration is the duration between Dtime and Ctime in seconds.

Table 2. An example of a ceased table.

Dtime	Ctime	duration (in seconds)	source	sourceID	errorID
2000/09/04 15:15:27	2000/09/06 00:03:08	18061	BSC	bsc22	q
2000/09/04 16:14:54	2000/09/09 16:55:52	434458	SITE	40833	s
2000/09/04 16:42:13	2000/09/09 11:37:15	413702	SITE	23004	c
2000/09/04 17:41:08	2000/09/08 09:22:04	333656	CELL	51821	i

3.2 Algorithm MSAP: Mining Sequential Alarm Patterns

After the data cleaning procedure, we transform the alarm data into a set of alarm sequences. Then, we devise an algorithm MSAP (standing for Mining Sequential Alarm Patterns) to extract useful sequential alarm sequences. Denote the set of errorID as E and define an alarm event as a pair (eid, t) , where $eid \in E$ is an errorID

and t is the time when the alarm event occurred. For example, an alarm event (A, 2001/2/10 16:14:08) means that the alarm A occurred at 16:14:08 on 10 Feb in 2001.

Definition 1: An alarm sequence S_{sid} is a sequence of ordered alarm events generated by the component sid , where sid is the sourceID and $S_{sid} = \langle (eid_1, t_1), (eid_2, t_2), (eid_3, t_3), \dots, (eid_n, t_n) \rangle$, where $t_i < t_j$ if $i < j$.

As described before, the alarm events are generated from individual components of the GSM. In this paper, we address the problem of mining sequential alarm patterns for the individual components and create an alarm sequence table containing alarm sequences. Table 3 is an alarm sequence table where the component of GSM is site (i.e., the base station) and the duration time of the alarm sequences is two hours, from 0:00am to 2:00am. Four alarm sequences (i.e., S_{2001} , S_{2002} , S_{2003} and S_{2004}) are given in Table 3. The *length* of an alarm sequence S_{sid} , denoted as $|S_{sid}|$, is the number of events in the sequence sid . A sequence of length k is called a k -sequence. It can be seen that the length of S_{2001} is four.

Table 3. An example of alarm sequences.

Site ID	Alarm sequences
2001	$\langle (A, 0:12) (C, 0:25) (D, 1:07) (C, 1:19) \rangle$
2002	$\langle (C, 0:37) (C, 0:42) (B, 1:11) (A, 1:36) \rangle$
2003	$\langle (B, 0:05) (C, 0:21) (C, 0:54) (A, 1:25) \rangle$
2004	$\langle (B, 0:27) (C, 0:37) (A, 1:22) \rangle$

The time difference between two events is in fact an important knowledge when mining the alarm data since, with this knowledge, one can predict the alarm sequence and take proper steps to prevent the occurrence of the alarms if all possible. In order to find the critical sequential alarm patterns concerned by network operators, algorithm MSAP allows network operators to set the restriction on the time difference between two alarm events in sequential alarm patterns. Thus, we have the following definition.

Definition 2: *Urgent window*, denoted as δ , is a user-specified time interval. We only count the pattern p that matches the candidate pattern $c \in C_2$, if the alarm events of p occur within the interval of an urgent window δ .

It will be seen that the size of an urgent window is set to determine the maximal time lag two events are allowed if they are to be viewed relevant to each other. With the set of the alarm sequences obtained, we next determine the sequential alarm patterns. Algorithm MSAP is devised and applied to alarm sequences to discover sequential alarm patterns. C_k is a set of candidate k -sequences. The occurrence count of C_k in alarm sequences is referred to as support. Let L_k represent the set of large k -sequences, where there are a sufficient number of alarm sequences containing this k -sequence. Such a threshold number is called minimum support in this paper. Using a candidate generation procedure, algorithm MSAP makes multiple scans over the set of alarm sequences to generate the possible sequential alarm patterns.

By observing the alarm sequences, the processing of counting support is nontrivial. For the customer transaction data, the support is usually only increased by one per customer even if the customer buys the same set of items in two different transactions. In contrast, alarm event A and alarm event C could occur several times in one day. As

a consequence, it is essential to develop the counting method to obtain the support of these candidate alarm sequences.

In order to find useful sequential alarm patterns, we only consider the event pair (i,j) , where i and j are alarm events and alarm j is the closest to alarm event i .

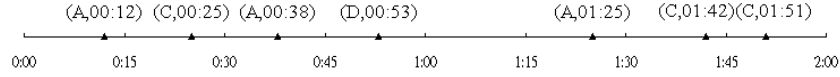


Fig. 4. An example of an alarm sequence.

Figure 4 shows the example of counting the support of $\langle AC \rangle$. Note that when we find an alarm event A, we only include for counting the alarm event C that is the closest to the alarm event A. Therefore, $\langle (A, 0:12) (C, 1:42) \rangle$, $\langle (A, 0:12) (C, 1:51) \rangle$, $\langle (A, 0:38) (C, 1:51) \rangle$, and $\langle (A, 1:25) (C, 1:51) \rangle$ are not counted by algorithm MSAP. Thus, in this example, the support of $\langle AC \rangle$ is three as shown in Figure 5.

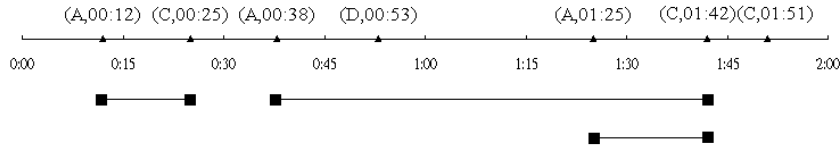


Fig. 5. The occurrence count of the sequential alarm sequence $\langle AC \rangle$.

With the above counting process, the large k -sequence has the following property:

Property 1: For any sequential alarm pattern $pattern_i = \langle r_1, r_2, \dots, r_k \rangle$ in L_k , the support of this sequential alarm sequence is smaller than that of the sequential alarm patterns $pattern_j = \langle q_1, q_2, \dots, q_{k-1} \rangle$ in L_{k-1} , where $q_x = r_x$ and $1 \leq x \leq k-1$. For example, the support of $\langle ABDF \rangle$ is smaller than that of $\langle ABD \rangle$.

In light of Property 1, we devise *Procedure Candidate_Generation* to generate the possible candidate k -sequences and algorithm MSAP to mine sequential alarm patterns. This solution procedure is outlined as follows.

Procedure Candidate_Generation

insert into C_k
select $p.event_1, p.event_2, \dots, p.event_{k-1}, q.event_1$
from $p \in L_{k-1}, q \in C_1$

Algorithm MSAP

Input: A set of alarm sequences $\{S\}$, urgent window δ , minimum support sup

Output: sequential alarm patterns

1. $L_1 = \{\text{large 1-sequence}\}$
2. $C_2 = \text{new candidates generated from } L_1 \text{ by calling } Procedure_Candidate_Generation$
3. **for** each alarm-sequence s in the database **do**

4. **for** each candidates $c \in C_2$ **do**
5. search c in the sequence s
6. **if** an event pair p matched $c \in C_2$ is found and the time-difference of event pair p is less than the *urgent window* δ
7. increase the count of $c \in C_2$ and record the time-difference
8. L_2 =candidates in C_2 with minimum support required
9. Compute the mean μ and standard deviation σ of the time-difference of each $l \in L_2$
10. **for** ($k=3$; $L_{k-1} \neq 0$; $k++$) **do**
11. **begin**
- C_k = new candidates generated from L_{k-1} by calling *Procedure Candidate_Generation*
12. **for** each alarm-sequence s in the database **do**
13. **for** each candidates $c \in C_k$ **do**
14. search c in sequence s
15. **if** a pattern x matched c is found and the time-difference of each event is less than or equal to δ
16. increase the count of the candidate $c \in C_k$
17. L_k =candidates in C_2 with minimum support required
18. **end**

Note that L_1 can be simply generated from the counting of appearing alarm events. Then, C_2 are obtained by *Procedure Candidate_Generation* (line 2 of algorithm MSAP). After generating possible C_2 , the occurrence count and the time difference of all candidate $c \in C_2$ are obtained by scanning all the alarm sequences (from line 3 to line 7 in algorithm MSAP). Algorithm MSAP calculates the mean and standard deviation of time-difference for each $l \in L_2$ (line 9 of algorithm MSAP). L_2 can be determined by proper trimming on C_2 . By utilizing *Procedure Candidate_Generation*, C_k can be generated by $L_{k-1} * C_1$ where $*$ is an operation for concatenation (from line 10 to line 18 in algorithm MSAP), which is very different from the one in the conventional sequential pattern mining. The support of each k -sequences is determined for the identification of L_k .

3.3 An Illustrative Example for Mining Sequential Alarm Patterns

Consider an example alarm sequence given in Table 3. In each iteration (or each pass), algorithm MSAP constructs a candidate set of large sequences, counts the number of occurrences of each candidate sequence, and then determines large sequences based on a pre-determined minimum support. In the first iteration, algorithm MSAP simply scans all the alarm sequences to count the number of occurrences for each alarm event. The set of candidate 1-sequence, C_1 , obtained is shown in Table 4a. Assume that the minimum support required is 2. The set of large

1-sequences, L_1 , composed of candidate 1-sequences with the minimum support required, can then be determined.

Table 4. Generation of candidate sequences and large sequences

(a) Generation of C_1 and L_1 .

C_1	support
A	4
B	3
C	7
D	1

→

L_1	support
A	4
B	3
C	7

(b) Generation of C_2 and L_2 .

C_2	support	time difference	μ	σ
AA	0		0	0
AB	0		0	0
AC	1	13	13	0
AD	1	55	55	0
BA	2	25, 55	40	15
BB	0		0	0
BC	2	16, 10	13	3
BD	0		0	0
CA	4	59, 54, 31, 45	47.25	10.64
CB	2	34, 29	31.5	2.5
CC	3	54, 5, 33	30.67	20.07
CD	1	42	42	0

→

L_2	support
BA	2
BC	2
CA	4
CB	2
CC	3

(c) Generation of C_3 and L_3 .

C_3	support
BAA	0
BAB	0
BAC	0
BAD	0
BCA	1
BCB	0
BCC	1
BCD	0

C_3	support
CAA	0
CAB	0
CAC	0
CAD	0
CBA	2
CBB	0
CBC	0
CBD	0

→

C_3	support
CCA	2
CCB	1
CCC	0
CCD	0

→

L_3	support
CBA	2
CCA	2

To discover the set of large 2-sequence, algorithm MSAP uses $L_1 * C_1$ to generate a candidate set of sequences C_2 . Next, the four alarm sequences in Table 3 are scanned.

The support of each candidate sequence in C_2 whose time difference between two alarm events is smaller than δ (i.e., the urgent window) is counted. Table 4b represents the result from such counting in C_2 . As described before, time plays an important role in mining sequential alarm sequences. Thus, algorithm MSAP calculates the mean and the standard deviation of the time difference for each 2-sequence in C_2 . The set of large 2-sequence, L_2 , is therefore determined based on the support of each candidate 2-sequence in C_2 . The set of candidate 3-sequence is generated from $L_2 * C_1$. Algorithm MSAP then scans the alarm sequences and discovers the large 3-sequences in Table 4c. Following the same procedure, algorithm MSAP discovers the large k -sequences. If there is no other large k -sequence discovered, algorithm MSAP stops and those large k -sequences are the sequential alarm patterns mined from the alarm data.

4 Experimental Results

The effectiveness of mining sequential alarm sequences is evaluated in this section. The alarm log data are provided by a cellular phone service provider. Experimental results of utilizing data cleaning techniques are shown in Section 4.1. Then, we present the experimental results of algorithm MSAP in Section 4.2.

4.1 The Impact of Data Cleaning

As mentioned before, the alarm log data contains not only useful information but also dummy information, thus calling for data cleaning. The execution time of mining sequential alarm patterns can also be reduced. We now examine the impact of data cleaning. We select one week alarm log data of site from the alarm database, where the number of sites is 3830. The performance improvement of MSAP is computed as follows.

$$\text{performance improvement} = \frac{|\text{execution time before cleaning} - \text{execution time after cleaning}|}{\text{execution time before cleaning}}$$

We then investigate the performance improvement when the operations of data cleaning are applied to the alarm data. Figure 7 shows the performance improvement while performing the procedure of data cleaning in advance. It can be seen that the execution times of algorithm MSAP vary due to the reason that the numbers of alarm events generated are different. Since the undetermined alarm events are substantial, occupying almost one-third of the alarm data volume before cleaning, the curve of the performance improvement ranges from 25% to 50%. For the process of merging repeated alarm events, the amount of repeated alarms is small. It can be seen in Figure 7 that the performance improvement is dominated by the operation of cleaning undetermined alarm events. Notice that the execution of algorithm MSAP after data cleaning is much faster than that without data cleaning, showing the merit of the data cleaning procedure.

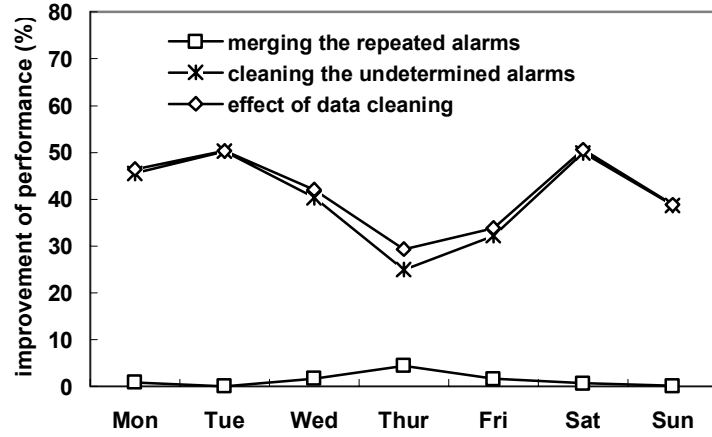


Fig. 7. The performance improvement while performing the operations of data cleaning.

4.2 Experimental Results of Algorithm MSAP

In this section, we evaluate the correctness of the sequential alarm patterns discovered by algorithm MSAP. Without loss of generality, we select one week alarm log data, and set the value of the minimum support to be 2000 and that of the urgent window to be one hour. Table 5 shows the selected sequential alarm patterns mined by algorithm MSAP and the descriptions of errorID is shown in Table 6. As can be seen in Table 5a, the time difference of two alarm events in L_2 is calculated by algorithm MSAP and is very important in predicting the network behaviors. For example, the sequential alarm pattern $\langle(g)(e)\rangle$ with its mean 815 and variance 962 is interpreted as when the problem of general line occurs, the loss of the frame alignment will likely occur after 815 seconds.

Table 5. The selected sequential alarm patterns discovered by algorithm MSAP.

(a) The selected large 2-sequences (L_2).

L_2	mean μ (in seconds)	standard deviation σ (in seconds)
$\langle(a)(b)\rangle$	803	593
$\langle(a)(c)\rangle$	965	944
$\langle(b)(d)\rangle$	1406	1202
$\langle(g)(e)\rangle$	815	962
$\langle(g)(f)\rangle$	1003	978

(b) The large 3-sequences (L_3).

<(a)(b)(a)>
<(a)(b)(c)>

Table 6. Descriptions of errorID.

errorID	Description
a	LPDL link down
b	LPDL link transition
c	Synchronization changed to holdover mode
d	End of holdover mode
e	Loss of frame alignment
f	Lower BER threshold exceeded
g	General line problems
h	Site Input Alarm detected

Table 7. The mining results obtained by algorithm MSAP with the value of the urgent window varied.

(a) The selected L_2 with the value of the urgent window to be one hour.

L_2	mean μ (in seconds)	standard deviation σ (in seconds)
<(b)(b)>	176	462
<(g)(e)>	826	980
<(g)(g)>	105	293
<(g)(f)>	962	946

(b) The selected L_2 with the value of the urgent window to be two hours.

L_2	mean μ (in seconds)	standard deviation σ (in seconds)
<(h)(h)>	480	1185
<(b)(b)>	240	720
<(b)(d)>	634	1308
<(g)(e)>	1813	2117
<(g)(g)>	132	468
<(g)(f)>	1759	1905

The impact of varying the values of the urgent window δ is next investigated. We select one day alarm log data and set the value of the minimal support to be 2000. The alarm sequences obtained are shown in Table 7. It can be seen from Table 7b that

more sequential alarm sequences (i.e., $\langle(h)(h)\rangle$ and $\langle(b)(d)\rangle$ are discovered by algorithm MSAP when the value of the urgent window becomes larger. This is due to the reason that with a larger value of δ , the support of alarm event pairs increases. The selection of the value of δ is determined by network operators and is also dependent upon the distribution of time difference between alarm event pairs.

It is worth mentioning that the sequential alarm patterns are beneficial for managing and designing alarm systems. With the sequential alarm patterns discovered, one can predict the alarm sequence and take the proper steps to prevent the occurrence of alarms if all possible. Furthermore, as pointed out in [11], alarm filtering requires the knowledge of the alarm sequences. By combining and transforming the related alarm events into one alarm, one is able to design proper alarm systems for telecommunication systems. Clearly, these sequential alarm patterns mined by our algorithm MSAP are also very useful in these design issues.

5 Conclusions

In this paper, we devised a solution procedure for mining sequential alarm patterns from the alarm data of a GSM system. First, by observing the features of the alarm data, we developed operations for data cleaning. After the data cleaning procedure, we transformed the alarm data into a set of alarm sequences. We devised a new counting method to determine the occurrence count of the sequential alarm patterns in accordance with the nature of alarms. By utilizing time constraints to restrict the time difference between two alarm events, we devised algorithm MSAP to obtain the mean and the standard deviation of two alarm events, and discover useful sequential alarm patterns. Algorithm MSAP is implemented and applied to test against a set of real alarm data provided by a cellular phone service provider. The quality of knowledge discovered is evaluated. The experimental results showed that the proposed operations of data cleaning are able to improve the execution of our mining algorithm.

Acknowledgement

The authors are supported in part by the Ministry of Education Project No.89-E-FA06-2-4-7 and the National Science Council, Project No. NSC89-2219-E-002-028 and NSC 89-2218-E-002-028, Taiwan, Republic of China

Reference

1. R. Agrawal, T. Imielinski, and A. Swami: Mining Associations between Sets of Items in Massive Databases. In Proceedings of ACM SIGMOD (May 1 993) 207—216.
2. R. Agrawal and R. Srikant. Mining Sequential Patterns. Proceedings of the Eleventh IEEE Inter-national Conference on Data Engineering (1995) 3—14.

3. B. Bruegge and B. Bennington. Applications of Mobile Computing and Communication. *IEEE Personal Communication* (February 1996) 64—71.
4. M-S. Chen, J. Han, and P. S. Yu.: Data Mining: An Overview from Database Perspective. *IEEE Transactions on Knowledge and Data Engineering* (December 1 996) (6):866—883.
5. N. Davies, G. S. Blair, K. Cheverst, and A. Friday: Supporting Collaborative Application in a Het-erogeneous Mobile Environment. *Computer Communication Special Issues on Mobile Computing* (1996).
6. M. H. Dunham: Mobile Computing and Databases. Tutorial of International Conference on Data Engineering (February 1998).
7. A. Elmagarmid, J. Jain, and T. Furukawa: Wireless Client/Server Computing for Personal Information Services and Applications. *ACM SIGMOD RECORD* (December 1995) 24(4):16—21.
8. Minos N. Garofalakis, Rajeev Rastogi, and Kyuseok Shim: SPIRIT: Sequential Pattern Mining with Regular Expression Constraints. In *Proceedings of VLDB* (September 1999) 223-234.
9. J. Han, G. Dong, and Y. Yin: Efficient Mining of Partial Periodic Patterns in Time Series Database. In *Proceedings of the 15th International Conference on Data Engineering* (March 1999).
10. J. Han, M. Kamber: *Data Mining: Concepts and Techniques*. Academic Press (2001).
11. K. Hätönen, M. Klemettinen, H. Mannila, P. Ronkainen, and H. Toivonen: Knowledge discovery from telecommunication network alarm databases. In *Proceedings of the Twelfth International Conference on Data Engineering*, New Orleans, Louisiana (1996) 115—122.
12. N. Krishnakumar and R. Jain: Escrow Techniques for Mobile Sales and Inventory Applications. *ACM Journal of Wireless Network* (July 1997) 3(3):235—246.
13. D. L. Lee: Data Management in a Wireless Environment. Tutorial of International Conference on Database System for Advance Applications (April 999).
14. H. Mannila, H. Toivonen, and A. I. Verkamo: Discovery of Frequent Episodes in Event Sequences. *Data Mining and Knowledge Discovery* (1996) 1(3):259—289.
15. M. Satyanarayanan: Mobile Information Access. *IEEE Personal Communication* (February 1996) 26—33.
16. Marek Wojciechowski: Interactive Constraint-Based Sequential Pattern Mining, *Proc. of the 5th East European Conference on Advances in Databases and Information Systems (ADBIS'01)*, Vilnius, Lithuania (2001), to appear.