Associative Data Mining for Alarm Groupings in Chemical Processes

Savo Kordic[1] Peng Lam[1] Jitian Xiao[1] Huaizhong Li[2]

[1]The School of Computer and Information Science, Edith Cowan University, Perth 6050, Western Australia
[2]School of Computer Science and Engineering, Wenzhou University Town, Zhejiang 325035, China

Abstract

Complex industrial processes such as nuclear power plants, chemical plants and petroleum refineries are usually equipped with alarm systems capable of monitoring thousands of process variables and generating tens of thousands of alarms which are used as mechanisms for alerting operators to take actions to alleviate or prevent an abnormal situation. Over-alarming and a lack of configuration management practices have often led to the degradation of these alarm systems, resulting in operational problems such as the Three-Mile Island accident. In order to aid alarm rationalization, this paper proposed an approach that incorporates a context-based segmentation approach with a data mining technique to find a set of correlated alarms from historical alarm event logs. Before the set of extracted results from this automated technique are used they can be evaluated by a process engineer with process understanding. The proposed approach is evaluated initially using simulation data from a Vinyl Acetate model. The approach is cost effective as any manual alarm analysis of the event logs for identifying primary and consequential alarms could be very time and labour intensive.

Keywords: Chemical plants, Data mining, Frequent episodes, Alarm logs

1. Introduction

Alarms are used as mechanisms for alerting operators to take actions that would alleviate or prevent an abnormal situation. Chemical plants such as petrochemical plants produce huge volumes of alarm data on a daily basis. As such industrial processes increase in size and complexity, the distributed control systems (DCS) that automate and monitor these processes have increased in complexity and have resulted in an ever increasing number of alarms being presented to the operators. In addition, a view of many process engineers has been one of adding alarm points to monitor anything that they have some doubts about since the addition has no actual dollar cost to the company. All it involves is adding alarm points to the existing DCS. This practice has undoubtedly resulted in the most common complaint – too many inappropriate alarms are generated in an emergency, thus making the alarm system very difficult to use in situations where it is most urgently needed. As an example, the alarm system during the 1994 incident at the Texaco Refinery, Milford Haven loaded the operators with one alarm every 2-3 seconds (20 – 30 alarms per minute) for a 5 hour duration leading to the explosion [1].

Although a chemical plant is “data rich”, few of them have the time, expertise or the appropriate tools to analyze and see how all the process variables as well as all the alarm tags are interrelated and whether there is a consistent pattern to what is happening. The use of statistics in the refining and petrochemical industries is a relatively new approach although it has been widely used in other areas such as product development, parts manufacturing and in research and development [2]. In the past, performing a manual analysis of such quantities of data would have been a very time consuming and intensive process. However, the availability of powerful computers nowadays means that these alarm events can be stored in a database, and off-line data analysis performed to assist not only in analysis of past plant alarm system performance, but also to predict future alarm system behaviour. The nature of process data (i.e. dynamic, multidimensional and voluminous) makes it very difficult for manual analysis as humans are weak in dynamic multi-dimensional analysis. This paper proposes a strategy that combines the processing powers of computers, domain heuristics and data mining algorithms to discover relationships and patterns hidden among the vast amounts of data. Unlike other approaches, the proposed technique uses a context-based data segmentation strategy which aids in producing results that address questions like “What is the group of alarm tags that were activated after TAG Q (alarm tag of interest) went into activation?”

It is important to detect the alarm groupings from chemical process data as they can then be used for alarm rationalization to remove redundant alarms,

identify bad actors and most importantly to re-establish an effective alarm system. The process data of a chemical plant has its life history; the idea is to extract information with minimal human interaction. The proposed approach is very cost effective as reams and reams of alarm data can be analyzed, and a set of grouped tags that are related in some manner is the output. The cost of computer time is very cheap compared to that of a process engineer. The process engineer will evaluate the results, consisting of groupings of related alarm tags - a very small set compared to the original data set. The re-alignment of the alarm system using the extracted information will reduce alarm floods in an abnormal situation. The main contribution of this paper is as follows:

1) A context-based segmentation technique that takes advantage of local event based segmentation and thus dramatically reduces the search space. In contrast to approaches like WINEPI [3] and MINEPI [4] that used fixed sized segmentation windows, the size of the windows used here are constructed adaptively.

2) A heuristic-based preprocessing strategy that incorporates domain specific concepts to remove spurious data points in a chemical process sense before the discovery of frequent itemset takes place. Since the number of co-related patterns extracted by frequent episode mining can be huge, the size of the resulting data set is reduced by this heuristic-based preprocessing strategy which reduces the total number of events within each transaction.

The rest of this paper is organized as follows: Section 2 describes related work and Section 3 introduces the proposed technique for finding groups of correlated alarm tags. The Vinyl Acetate process, experiments and relevant results are presented in Section 4 and lastly the conclusion is found in Section 5.

2. Related work

A related research area is alarm pattern discovery in telecommunications networks. Hatonen et al. [5] have used frequently occurring episodes for discovering patterns in alarm databases to predict faults. Their subsequent work TASA (Telecommunication Alarm Sequence Analyzer) [6] was a system for mining knowledge from telecommunications networks in terms of 'episode rules'. An episode [7] is defined as a partially ordered collection of events that occur together within a time window. While a serial episode occurs in a fixed order, a parallel episode is an unordered collection of events (i.e. trivial partial order). TASA used WINEPI and MINEPI for discovery of episode rules - WINEPI from a given class of episodes by sliding a fixed size window over the input sequence, and MINEPI according to minimal occurrences of episodes. One of the main difficulties when analysing alarm sequences in WINEPI was to specify the window size within which an episode must occur. If the window is too small alarm information will be lost, or if the window is too big, then the efficient mining of large numbers of distinct alarm types will be an increasingly difficult process. WINEPI decomposes each sliding window to elementary episodes in the episode recognition phase, so clearly this approach will not be useful for mining very large numbers of alarm tags, unless the window width is very short. In contrast, MINEPI does not rely on the sliding window strategy, instead, it counts minimal occurrences of episodes within the given time bounds. Accordingly, for practical mining purposes, relatively small time intervals need to be given in order to reduce the search space, otherwise, the total number of both candidate and frequent episodes could grow dramatically. Not surprisingly since both WINEPI and MINEPI employ the candidate generation method (i.e. an Apriori-like algorithm) to find frequent episodes, they could encounter difficulties when mining long sequential patterns with many distinct alarm types.

Another approach for mining such patterns is to use the frequent pattern growth technique [8] which mines frequent patterns without candidate generation. The method is based on the FP-Tree data structure. However, when a database is too large or has many diverse alarm types it will be difficult to store in a main memory or to construct an FP-tree. To solve this problem, PrefixSpan [9] uses prefix-based projections to reduce the size of sequential databases. CloSpan [10] reduces the number of possible frequent patterns by mining closed sequential patterns in large datasets and more recently, BIDE [11] mines frequent closed sequences avoiding a candidate generation procedure. Others like SPADE [12] use equivalence prefix classes to decompose the search space and incorporate lattice-based search strategies. However, most of the existing sequential data mining approaches for handling telecommunications data (e.g. GSP [13]) have their historical origin in the Apriori technique and thus suffer from candidate generation related problems as well as generating an enormous number of candidate frequent episodes or episode rules from very high dimensional alarm data, usually consisting of several hundreds or even thousands of distinct alarm tags. Klemettinen et al. [14] introduced the concept of templates in an attempt to address the large number of mined rules and Devitt and Duffin [15] used network topology to evaluate mined alarm sequence plausibility and reduce the set of mined sequences by eliminating those that violated the network topology.

Approaches to address the issue of alarm management in industrial processes such as petrochemical, pulp industries and steel mills include the concept of alarm sanitation [16], alarm cleanup toolbox [17], the application of alarm handling algorithms which are applied to Multilevel Flow Models (MFM) [18] and techniques related to Fault Detection and Isolation (FDI). FDI approaches involved using specifically designed models based on laws of physics which are obtained through a linearization process. The development of such models is very time consuming and any subsequent changes to the process require modifications to the existing models. Neural networks are increasingly used in this approach to obtain a model from the data which can be used to carry out fault diagnosis. Similarly the MFM models representing the intended functionality of a chemical process must also be constructed first before any application of alarm handling algorithms and thus suffer from the same limitations as FDI approaches. In addition, both of these approaches do not use the historical process data. Alarm sanitation and the alarm cleanup are related approaches that detect badly tuned alarm points automatically and help process engineers in re-tuning the alarm limits. The alarm cleanup toolbox incorporates the use of signal processing techniques and LARA, a rule-based expert system classifier which classifies continuous process data based on their statistics. The aim of the approach is to examine historical process data off-line and to suggest changes to existing alarm settings for those alarm tags identified as being badly tuned. Being a rule-based system makes this approach very much dependent upon the expertise and knowledge of the human experts as well as the labour involved in coding up the rules.

3. Proposed approach

The overall algorithm involving four phases is shown below. It is based on a context-based segmentation strategy (i.e. a transaction has a clear contextual meaning as the whole data set can be segmented into *n* transactions while *n* is the activation number of an alarm tag), and incorporates some associative mining techniques.

Algorithm:

Input: sequence of historical alarm event logs. An alarm sequence is a collection of alarm events which includes alarm activation and alarm return types.

Extract the relevant information associated with alarm tags from event log-file and put into an appropriate format. (*Phase 1)*

For all alarm tags in the event log file

Extract the $W_{A\text{-}R(i)}$ and the $W_{R\text{-}A/TW(i)}$ sets of transactions for each alarm tag (*Phase 2*)

For each alarm tag

Do the filtering based on Return-point (R-p) strategy or Activation-point (A-p) strategy (*Phase 3*)

Using the R-p data or the A-p data, do "frequent maximal itemset mining" to obtain a set of co-occurring alarm tags associated with each tag of interest (*Phase 4*)

3.1. Phase 1: data preparation

As different vendors of modern control systems generate their own format for messages, the format of alarm data is both software and hardware dependent. This phase extracts a sequence of alarm events from an alarm event log file. As a minimum requirement, the alarm event log consists of records with fields that store information for a unique identifier for each alarm tag, time/date, alarm priorities, alarm settings and the possible states of an alarm tag [ACTIVATION, RETURN]. When an alarm tag is in ACTIVATION state, it implies that the value of the associated process variable is outside its normal operational setting, and when this value returns to the normal operational settings, the alarm tag is then in a RETURN state. Each extracted event has only three attributes – alarm tag identifier, its state and time of the event occurrence.

Using a simple algorithm, the event log file is processed to produce formatted alarm data as shown in Table 1. The symbol A stands for activated and R for returned. The approach captures the first time instance an alarm tag goes into activation and the time instance the alarm tag has been returned. For example, in the column associated with TAG 1 in Table 1, it shows that TAG 1 was activated at 9:32 and it returned at 9:35, the duration of this tag being in an activation state was 3 minutes. TAG 1 is re-activated at 9:37. Similarly, TAG 6 was activated at 9:32 and it returned at 9:36. The collection of alarm activations/returns and time-stamps associated with a defined set of alarm tag identifiers as shown in Table 1 forms an alarm sequence.

Time	TAG						
Interval (every 1 min)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
1 9:32	*A* 1					*A* 6	*A* 7
2 9:33				*A* 4			*R* -7
3 9:34		*A* 2					
4	*R*						

9:35	-1						
5 9:36		*R* -2				*R* -6	
6 9:37	*A* 1						
7 9:38			*A* -3	*R* -4	*R* -5		*A* 7
8 9:39	*A* -1						

Table 1: An example of alarm sequence.

3.2. Phase 2: extracting context-based transactions

Instead of using a fixed sized sliding window (as in WINEPI), two types of windows are used here. The size of the first type of window used in this phase is adaptively defined as the duration of the tag of interest in activation state (denoted as $W_{A\text{-}R}$). The starting point of the window is defined as the time for the tag of interest going into activation and the ending point is where it goes into the corresponding return state. For example, if the tag of interest is TAG 1 in Table 1, the windows are defined as $W_{A\text{-}R(1)}$, from 9:32 to 9:35 and $W_{A\text{-}R(2)}$ from 9:37 to 9:39. In the case of TAG 6, its $W_{A\text{-}R(1)}$ window is defined from 9:32 to 9:36. A second type of window (denoted as $W_{R\text{-}A/TW}$) used here is defined by a fixed user-defined duration after a tag of interest moves back into a return state (after being activated). This user-defined duration may be shortened if the tag of interest is reactivated. Again, using a user-defined window (W_{ud}) of 3 minutes and TAG 1 as an example, its $W_{R\text{-}A/TW(1)}$ should start from 9:35 and end 3 minutes later at 9:38 but as TAG 1 is re-activated again at 9:37 this window will terminate at the point TAG 1 is re-activated (i.e. a duration of 2 minutes instead of 3 minutes). The value of W_{ud} is related to the process lag found in a chemical process and thus is a value that is provided by the domain experts (in this case the process engineer). In the case of a very fast process this value may be in seconds, otherwise in a slow process it may be in the order of minutes or hours.

At each point in time, the window is shifted along to the next activation instance of the tag of interest. Thus if the total number of activations of a specified alarm tag is *n*, then the whole alarm sequence can be segmented into *n* windows of $W_{A\text{-}R}$. Similarly the whole alarm sequence can be also segmented into *m* windows of typed $W_{R\text{-}A/TW}$ if there are *m* returns associated with the tag of interest. Obviously, this segmentation is an event-based extraction with a clear contextual meaning. Each of the *n* windows of $W_{A\text{-}R}$ captures data that indicates which other alarm tags also went into activation while the alarm tag of interest is in activation. For example, the transaction data in the $W_{A\text{-}R(1)}$ for TAG 1 would be [1 2 4 6 7] as TAG 2, 4, 6 and 7 also went into activation while TAG 1 is in an activation state. On the other hand each of the *m* windows of $W_{R\text{-}A/TW}$ captures data that showed which other alarm tags also return within a user-defined window after the tag of interest returned. Using TAG 1 as an example, the transaction data in its $W_{R\text{-}A/TW(1)}$ is [-1 -2 -6] as TAG 2 and 6 also returned in that window. At the end of this phase, the alarm sequence is processed for all alarm tags of interest and for each alarm tag, there are two sets of transactions each associated with the two types of defined windows respectively (e.g. Column 1 and 2 of Table 2).

3.3. Phase 3: transaction filtering

Two strategies, incorporating chemical process related heuristics, were used in this phase for filtering transaction data from Phase 2. These are:

- **Return-point strategy** (R-p) – This strategy makes use of the *cause-effect* relationships found in phenomena such as chemical processes. The rationale used here is that a *dependent* variable (alarm tag) should return after the *independent* variable (i.e. a cause alarm) has returned. If the cause of the problem is resolved then all subsequent alarm tags related to the problem should return some time after the initial alarm tag associated with the cause has returned.
- **Activation-point strategy** (A-p) – The heuristic used here relates to activations of alarm tags after the cause of the problem has been resolved. The rationale is that dependant variables (*consequence alarms)* once returned, should not re-activate after the *independent* variable is returned (i.e. a cause alarm is eliminated).

The two sets of transactions associated with each alarm tag are processed using one of the above strategies to produce a set of filtered transactions. Table 2 shows an example of processing transactions associated with TAG 3 using the Return-point strategy and a W_{ud} of 30 minutes. The resulting transactions in column 3 of Table 2 showed alarms tags found in the transaction from a $W_{A\text{-}R(i)}$ window which also returned after the tag of interest returned.

$W_{A\text{-}R(i)}$ of TAG 3	$W_{R\text{-}A/TW(i)}$ of TAG 3 time window = 30 min	Co-occurring Tags found in $W_{A\text{-}R(i)}$ and $W_{R\text{-}A/TW(i)}$ of TAG 3
3 7 6 9 11	-3 -7 -9	3 7 9
3 7 6 9 11	-3	3
3 5 7 9	-3 -7	3 7
3 7 6 9 11	-3 -7 -11	3 7 11

Table 2: An example of processing transactions associated with TAG 3 using the Return-point strategy and a W_{ud} of 30 minutes.

Table 3 shows an example of processing transactions associated with TAG 3 using the Activation-point strategy and a W_{ud} of 30 minutes. As can be seen from Column 2 of this table, the strategy looked at tags which were activated after the tag of interest (i.e. TAG 3) returned. Alarm tags found in the transactions from a $W_{R\text{-}A/TW(i)}$ window are then eliminated from the corresponding transaction from a $W_{A\text{-}R(i)}$ to generate the resulting transactions shown in Column 3 of Table 3.

$W_{A\text{-}R(i)}$ of TAG 3	$W_{R\text{-}A/TW(i)}$ of TAG 3 time window = 30 min	Tags found in $W_{A\text{-}R(i)}$ but NOT in $W_{R\text{-}A/TW(i)}$ of TAG 3
3 7 6 9 11	-3 9	3 7 6 11
3 7 6 9 11	-3	3 7 6 9 11
3 5 7 9	-3 8 4	3 5 7 9
3 7 6 9 11	-3 9 11	3 7 6

Table 3: An example of processing transactions associated with TAG 3 using the Activation-point strategy and a W_{ud} of 30 minutes.

3.4. Phase 4: frequent episode discovery

The required inputs to this stage are the outputs from Phase 3 and a user-defined support value (*sup*) which is in the range of 0 – 100%. After the series of steps from Phase 1 to Phase 3, the raw alarm data was segmented into sets of transactions associated with each specific alarm tag and either the R-p or A-p filtering strategy. By the nature of the process involved in extracting these context-based transactions, a one-level (cause => depended group) frequent serial episode rule can be obtained via this phase, providing an answer to questions like: " What is the group of tags that were activated after TAG Q (tag of interest) went into activation?" (i.e. (TAG Q => [group of co-occurring alarm tags]). A domain expert such as a process engineer can decide on the *sup* value to be used in an exploratory manner. By increasing this value close to 100%, the number of possible one-level frequent serial episode rules can be reduced, thus ensuring that the extracted relationship is found in nearly all the transactions.

In order to find the groups of co-occurring alarm tags, a modified FP-growth algorithm has been implemented to output maximal frequent itemsets (i.e. the smallest frequent itemsets) that are associated with each alarm tag of interest. Note that a frequent itemset is defined as being a maximally frequent if none of its immediate supersets are frequent [19]. At the end of this phase, the list of all extracted frequent maximal itemsets associated with each alarm tag of interest (e.g. Table 4) are displayed for the domain expert to further refine the mining results as well as to validate the extracted rules using his/her domain knowledge. Based on a set of associated constraints (i.e. filtering strategy = R-p or A-p, W_{ud} = 30 minutes, *sup* = 90% and confidence = 100%), results obtained can be:

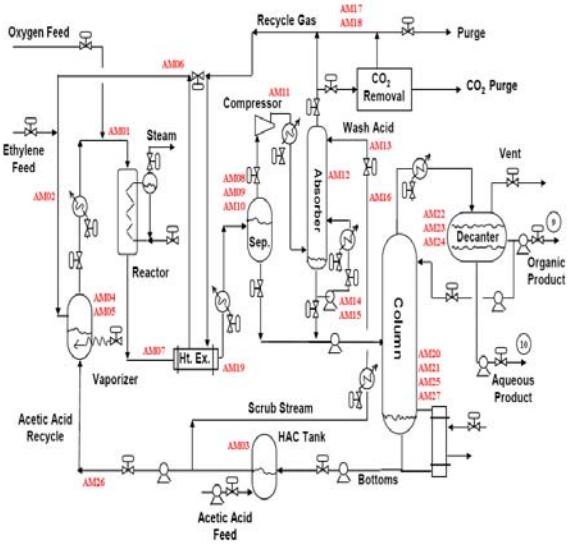
- an empty set associated with a specific tag of interest (e.g. TAG 1), thus indicating that there are no co-occurring alarms tags associated with this specific tag
- a non-empty set associated with a specific tag of interest (e.g. TAG 3), thus indicating that there are co-occurring alarms tags associated with this specific tag. The domain expert may validate this result using their domain knowledge and then carry out further investigation involving tags of this group. A point to note is that it is possible to have multiple tags in the set instead of just one (i.e. 16) tag in the set of frequent closed itemsets.
- If an alarm tag is associated with a process variable that's fixed then the result shown would be similar to that of TAG 26.

The significance of this phase is that only non-empty sets are of interest and the process engineers can carry out further validation or investigation before using the groupings for alarm rationalisation.

4. Experimentation

The proposed approach was evaluated using simulated data produced from a Matlab model of the Vinyl Acetate chemical process. Figure 1 shows the process flowsheet with the locations of 27 associated alarm tags denoted in the figure as AM01 to AM27. They are placed to measure the outputs of 27 controlled variables. In order to generate discrete alarm data using the Matlab model, it is necessary to perform a simulation twice. The first iteration of the simulation is performed to obtain the measurement outputs of the Vinyl Acetate model under normal operating conditions. Then faults are injected at random points in the second iteration of the simulation to obtain the disturbed measurement outputs of the model. The difference between the normal measurement outputs and the disturbed measurement outputs can then be used to generate discrete alarm data which are associated with the injection of disturbances.

Fig. 1: Vinyl Acetate process flowsheet showing location of the simulated alarm monitors (modified from [20]).



4.1. Generation of the simulated

4.2. Alarms

Using the normal process output stored in the first data file and the disturbed process output stored in the second data file, it is straightforward to obtain the changes in the process measurements which are caused by the injected disturbances. For simplicity, it is assumed that a simulated alarm is activated if the following condition is satisfied:

Abs((Disturbed output magnitude – Normal output magnitude)/normal output magnitude) ≥ Sensitivity of the simulated alarm monitor

The simulated alarm returns to normal if the above condition is not satisfied. Note that the sensitivity in the above condition refers to the signal detection sensitivity of a measurement instrument. The sensitivity for all simulated alarm monitors is 0.0005. Two sets of alarm data described below, each associated with a different fault were produced. Unlike process data, these alarm data are discrete (i.e. 0s and 1s).

Simulated fault data set 1

Frequent disturbances are injected into the Vinyl Acetate process to induce a response in measurement outputs. Only one type of disturbance is introduced in this simulation, namely the *loss of the fresh HAc feed stream* (associated with alarm TAG AM03)**,** with each fault lasting over different durations. The fault was injected 10 times. The measurement outputs are monitored and sampled at a frequency of *1 sample in five seconds*.

Simulated fault data set 2

Frequent disturbances of one type, namely the *loss of %O2 in the Reactor Inlet* (associated with alarm TAG AM01), are injected into the Vinyl Acetate process to induce a response in measurement outputs. Each injected fault lasted for different durations. The fault was also injected 10 times. The measurement outputs are monitored and sampled at a frequency of *1 sample in one second*.

4.3. Initial results and discussion

Table 4 and 5 respectively show the grouping results where the transactions associated with the two sets of different simulated fault data were processed via the proposed technique using *sup* = 90% confidence = 100% and W_{ud} = 30 minutes and both filtering strategies. The first column in these two tables identifies the alarm tags; the second and third columns show the associated result sets of co-occurring alarm tags obtained from mining the transactions obtained from the A-p and R-p strategies respectively. Table 4 shows the result sets associated with the complete set of 27 alarm tags and Table 5 only shows alarm tags where the result set is not empty.

By examining entries like those found in Table 4 and Table 5, a process engineer can very quickly identify initial alarm groupings that are of interest (i.e. non empty sets). The process engineer can then use the identified groups to further examine the data. In comparison, when the same data set was mined using WINEPI, using a fixed size window of 10 minutes, it was shown that at *sup* = 90%, only one result sets were obtained and at a low support level, the result set consists of tens of thousands of rules, thus making it extremely difficult to analyse.

TAG	Activation-Points Preprocessing	Return-Points Preprocessing
1 %O2	1 => ( )	1 => ( )
2 Press	2 => ( )	2 => ( )
3 HAc-L	3 => (5 2 12 16 9 4 18 6 11 8 14)	3 => (16)
4 Vap-L	4 => (25)	4 => ( )
5 Vap-P	5 => ( )	5 => ( )
6 Pre-T	6 => ( )	6 => ( )
7 RCT-T	7 => ( )	7 => ( )
8 Sep_L	8 => ( )	8 => ( )

9 Sep-T	9 => (27 18)	9 => ( )
10 Sep-V	FIXED	FIXED
11 Com-T	11 => ( )	11 => ( )
12 Abs-L	12 => ( )	12 => ( )
13 Cir-F	FIXED	FIXED
14 Cir-T	14 => ( )	14 => ( )
15 Scr-F	FIXED	FIXED
16 Scr-T	16 => (18)	16 => ( )
17 %CO2	17 => ( )	17 => ( )
18 %C2H6	18 => 14	18 => ( )
19 FEHE-T	19 => ( )	19 => ( )
20 %H2O	20 => ( )	20 => ( )
21 Col-Ts	no alarms	no alarms
22 Org-L	no alarm	no alarms
23 Aqu-L	23 => ( )	23 => ( )
24 Col-L	24 => ( )	24 => ( )
25 Vap-In	25 => ( )	25 => ( )
27 %VAc	Final Product	Final Product

Table 4: Results associated with sets of co-occurring alarm tags obtained from the application of the proposed approach using Simulated Fault data **Set 1** with respect to min support = 90%, time window = 30 min, and confidence = 100%.

The result set associated with alarm tag **3 HAc-L** in Table 4 has the biggest number of co-occurring alarm tags associated with it when the A-p strategy is used but has only one co-occurring alarm tag for the case of the R-p. This is due to the fact that R-p strategy emphasis is on the alarm tags also returning within a time window of size N after the tag of interest has returned. On the other hand, in the case of A-p, the focus has been on alarms that are re-activated after the tag of interest has return, the rationale being if the causal alarm tag has returned, related alarm tags should not be re-activated unless they are nuisance alarms. Using this rationale, such alarm tags found in the $W_{A\text{-}R(i)}$ are removed. The larger result set implies that those alarms tags have not been re-activated after the tag of interest returned and are most likely related to the tag of interest.

In terms of checking whether the groupings are correct, the result sets associated with each tag can be checked against the process flowsheet in Figure 1. When there's a *loss of the fresh HAc feed stream*, alarm TAG AM03 goes into activation**,** and by following the flow of the directional lines, it can be seen that the next alarm Tag likely to go into activation would be AM16. In fact Tag 16, being close to Tag 3, would return within a time frame of 30 minutes (as seen from the result associated with the R-p strategy). In terms of the set 3 => (5 2 12 16 9 4 18 6 11 8 14), by looking at the flowsheet it is possible to have these tags going into activation while Tag 3 is in activation. A tag like 15 has not been picked up because it is FIXED.

Similarly, the results in Table 5 associated with mining the second set of data can be validated against the flowsheet. The fault is associated with *loss of %O2 in the Reactor Inlet* (AM01) and again it can be seen from the flowsheet that after Tag 1 goes into activation, Tag 7 and Tag 19 would go into activation. The reason that Tag 7 is not in the result sets is because it is chattering (i.e. going in and out of activation) because of its setting. Again the R-p strategy of filtering of data has a smaller grouping due to the fact that only alarm Tag 19 has returned in the given time frame.

TAG	Activation-Points Preprocessing	Return-Points Preprocessing
1 %O2	1 => (27 9 18 11 4 19 6 12 8 20)	1 => (19)
4 Vap-L	4 => (12 6 19)	4 => ( )
6 Pre-T	6 => (12)	6 => ( )
7 RCT-T	7 => (12)	7 => ( )
11 Com-T	11 => (19)	11 => ( )
12 Abs-L	12 => (22)	12 => ( )
18 %C2H6	18 => (23 17)	18 => ( )
19 FEHE-T	19 => (12 23)	19 => ( )
25 Vap-In	25 => (3)	25 => ( )

Table 5: Results associated with the non-empty sets of co-occurring alarm tags obtained from the application of the proposed approach using Simulated Fault data **Set 2** with respect to min support = 90%, time window = 30 min, and confidence = 100%.

A point to note is that the process engineer can vary the value of minimum support (*sup*), and the size of $W_{ud,}$ to explore the alarm groupings space. By using a bigger value of the size of W_{ud} or by lowering the value of minimum support, the alarm grouping space to be explored can be widened. At the same time by using a smaller value of the size of W_{ud} or by choosing the value of *sup* to be closer to a value of 100, the alarm grouping space to be explored can be made smaller.

5. Conclusions

This paper has described an approach that incorporates a context-based segmentation approach with a data mining technique to find a set of correlated alarms from historical alarm event logs. The approach allows an exploration of the alarm grouping space to find groups of co-occurring alarms which can be used to aid alarm rationalisation. Before the set of extracted results from this automated technique are used they can be evaluated by a process engineer with process understanding. The proposed approach is evaluated initially using simulation data from a Vinyl Acetate model. Future work would involve data containing multiple types of different faults, simulated from the

Vinyl Acetate model as well as alarm data from petrochemical industries. The approach is more cost effective that any manual alarm analysis of the event logs for identifying primary and consequential alarms which are very time and labour intensive.

References

[1] Health & Safety Executive Report, The explosion and fires at the Texaco Refinery, Milford Haven, 24 July 94, published by the HSE C30, 1997.

[2] D. Stockill, Unlock the Value in Process Data - Making Money with Statistics and Data Mining. *Honeywell IC Users Group Symposium*, 2001.

[3] H. Mannila, H. Toivonen and A. I. Verkamo, Discovering frequent episodes in sequences. *Proc. of the First International Conference on Knowledge Discovery and Data Mining (KDD '95)*, pp. 210-215, 1995.

[4] H. Mannila, H. Toivonen, Discovering generalized episodes using minimal occurrences. *Knowledge Discovery and Data Mining*, pp. 146-151, 1996.

[5] K. Hätönen, M. Klemettinen, H. Mannila and P. Ronkainen, et.al., Knowledge Discovery from Telecommunication Network Alarm Databases. *Proc. of the twelfth International Conference on Data Engineering*, pp. 115-122, 1996.

[6] K. Hatonen, M. Klemettinen, H. Mannila and P. Ronkainen, et.al., TASA: Telecommunications Alarm Sequence Analyzer, or How to enjoy faults in your network. *IEEE Network Operations and Management Symposium (NOMS'96)*, pp. 520-529, 1996.

[7] H. Mannila, H. Toivonen, A. I. Verkamo, Discovery of Frequent Episodes in Event Sequences. *Data Mining and Knowledge Discovery*, 1: 259-289, 1997.

[8] J. Han, J. Pei, Y. Yin, Mining Frequent Patterns without Candidate Generation. *Proc. of the 2000 ACM SIGMOD International Conference on Management of Data*, pp. 1-12, 2000.

[9] J. Pei, J. Han, B. Mortazavi-Asl and H. Pinto, PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. *Proc. of the 17th International Conference on Data Engineering (ICDE'01)*, pp. 215-224, 2001.

[10] X. Yan, J. Han, R. Afshar, CloSpan: Mining closed sequential patterns in large datasets. *Proc. of the 3rd SIAM Int. Conf. Data Mining*, pp.166-177, 2003.

[11] J. Wang, J. Han, BIDE: Efficient Mining of Frequent Closed Sequences. *Proc. of 20th International Conference on Data Engineering (ICDE'04)*, pp. 79-90, 2004.

[12] M. J. Zaki, SPADE: An Efficient Algorithm for Mining Frequent Sequences. *Machine Learning Journal*, 42: 31-60, 2001.

[13] R. Srikant, R. Agrawal, Mining Sequential Patterns: Generalizations and Performance Improvements. *Proc. of 5th International Conference on Extending Database Technology (EDBT)*, pp. 3-17, 1996.

[14] M. Klemettinen, H. Mannila, H. Toivonen, Interactive Exploration of Discovered Knowledge: A Methodology for Interaction, and Usability Studies. *Technical Report C-1996-3, University Of Helsinki, Department of Computer Science, 1996.*

[15] A. Devitt, J. Duffin, and R. Moloney, Topographical proximity for mining network alarm data, *Proc. of the 2005 ACM SIGCOMM Workshop on Mining Network Data*, pp. 179-184, 2005.

[16] J. E. Larsson, Simple Methods for Alarm Sanitation. *Proc. of IFAC Symposium on Artificial Intelligence in Real-Time Control*, 2000.

[17] J. Ahnlund, T. Bergquist, Alarm Cleanup Toolbox. *M.Sc. Thesis, Department of Information Technology, University of Lund, Lund, October, 2001*.

[18] J. E. Larsson, Diagnostic Reasoning Based on Mean-End Models: Experiences and Future Prospects. *Knowledge-Based Systems*, 15 (1-2): 103-110, 2002.

[19] D. Burdick, M. Calimlim, and J. Gehrke, MAFIA: A Maximal Frequent Itemset Algorithm for Transactional Databases. *IEEE Transactions on Knowledge and Data Engineering*, 17(11): 1490-1504, 2005.

[20] R. Chen, K. Dave, T. J. McAvoy and M. Luyben, A Nonlinear Dynamic Model of a Vinyl Acetate Process. *Ind. Eng. Chem. Res.*, 42(20): 4478-4487, 2003.