Date of publication SEP-30, 2022, date of current version JUL-28, 2022.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.21.3.2689

Alarm Pattern Recognition in Continuous Process Control Systems using Data Mining

CHETANA BELAVADI, VANDANA SUDHAKAR SARDAR, SHILPA SHASHIKANT CHAUDHARI

Ramaiah Institute of Technology, Bangalore, Karnataka, India.
(E-mail: cbelavadi5@gmail.com, vandana.s@msrit.edu, shilpasc29@msrit.edu)

Corresponding author: Chetana Belavadi (e-mail: cbelavadi5@gmail.com).

ABSTRACT An alarm management system with the Human Machine Interface in a process control system is used to alert an operator of any abnormal situation, so that corrective action can be taken to ensure safety and productivity of the plant and quality of the product. An alarm system reporting many alarms even during the normal state of the plant is due to chattering alarms, duplicated alarms, intermittent equipment problems and certain alarms configured in the system which may not have any importance. In such a situation the operator may miss certain critical alarms leading to undesirable outcomes. So, to have an optimum alarm system, the unwanted alarms have to be identified and eliminated. In this paper, we propose an offline method to identify repetitive, frequent sequences or patterns using PrefixSpan and Bi-Directional Extension algorithms. With the identified sequences or patterns, plant operation experts can improve the effectiveness of the alarm system through alarm rationalization so that this will help the operator in making the plant more safe, reliable and productive. The main objectives of this work are the following: (i) to use a definitive method to represent alarm data in an alarm log which is Temporal data as Itemsets without a need for complex mathematical, statistical or visual methods; (ii) to use data mining algorithms for identifying Frequent sequences which can be implemented on a normal computing resource such as Personal computer; (iii) to apply the method to the complete alarm data available no matter how big they are; (iv) to study and establish that the chosen method is possible to be applied to larger sized datasets.

KEYWORDS alarm management; alarm flood; alarm data analysis; alarm pattern recognition; frequent pattern.

I. INTRODUCTION

PROCESS control systems are designed to function as pieces of equipment along the production line. They use different methods to measure and control the process during manufacturing, to return data for useful purposes. Minor faults propagation in the system causes equipment damage, production loss, safety hazards and impact on the environment. Alarm system for minor to major fault indication can prevent the faults and failures in the process. Alarm system consists of hardware and software component to trigger an alarm. Alarm is triggered due to equipment malfunction, process deviation, or abnormal condition and requiring operator response [1]. Alarm can be recorded in database for further analysis and action as well as indicated by audio/visual means. Alarm databases maintain information about each alarm in terms of time at which it is generated, name, identifier, tag, priority, location/unit where it occurred etc. Low cost alarm can be easily added, removed or reconfigured. Chattering alarms transits into and out of an alarm status in a short interval of time, or sometimes even multiple times per minute [1]. Duplicate alarms are of two types – (1) Dynamic Duplicate Alarms tend to occur when a process event leads to multiple alarm annunciations in different ways. (2) Configured Duplicate Alarms tend to occur because of incorrect Distributed Control Systems (DCS) alarm settings, leading to duplicate alarm configurations [1].

These days, it has become a very common problem in industrial plant control systems, that the alarm screens keep scrolling in real time due to many alarms being reported, some of which may not be of any value to the operator, thereby defeating the purpose of alerting the operator. The number of alarms an operator receives per hour is in the magnitude of tens, hundreds or even thousands. A majority of these alarms are false or nuisance alarms also. An alarm system reporting many alarms even during the normal state of the plant is due to chattering alarms, duplicated alarms, intermittent equipment problems and certain alarms configured in the system which may not have any importance. Too many such alarms only

distract the operator from operating the plant, which can cause critical alarms to be ignored, and lead to distrust of the alarms by operators. So, it is important to identify these unwanted alarms and eliminate them so that the operator efficiency is improved.

The existing work focuses on the various tools and techniques available for alarm pattern recognition and classification [2-8] using complex techniques but does not throw light on the way frequent pattern mining or data mining algorithms can be applied to find frequent sequences or patterns in alarm data. The existing work is more on the study of techniques and algorithms which help in classifying alarm floods online or find patterns in them and also help in the graphical visualization of these alarm floods [9-21]. As part of the present work, the main aim is to propose an offline method to help plant operators or experts to identify repetitive sequences or patterns in the alarms presented to them using data mining techniques to further assist in the area of alarm rationalization. The objective would be to see whether common frequent pattern mining algorithms can be applied to this problem and find ways/methods to accurately represent temporal data as itemsets. The selected/studied algorithms should also be able to be applied effectively to large sized datasets as alarm data analysis over larger periods of time can consist of tens of thousands or hundreds of thousands of data.

The rest of the paper is organized as follows. Section II presents the preliminaries on alarm systems along with tools and techniques used. Section III focuses on identification of repetitive, frequent sequences or patterns in alarm system using PrefixSpan and Bi-Directional Extension algorithms to further assist in the area of alarm rationalization. Section IV demonstrates the effectiveness of the proposed offline method to help plant operators. Section V provides the concluding remarks and future work.

II. RELATED WORKS

Studies on alarm pattern recognition in modern data process plants exist in the literature. The various papers referred are broadly classified into three types as shown in Fig. 1. (1) Online alarm flood classification, (2) Alarm flood pattern/sequence recognition/sequence mining and (3) Visualization of alarm data.

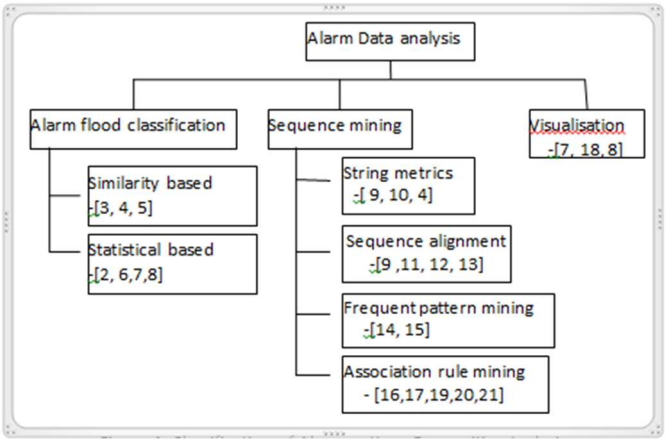

Figure 1. Classification of Alarm pattern Recognition techniques

Alarm flood classification classifies an incoming alarm flood on the basis that this new alarm is matched with a set of previously occurred alarm floods, both analogous to the same anomalous situation. Alarm flood classification is done in two stages: (1) Detecting stage detects/determines whether the alarm flood being classified belongs to a new class or not; (2) Classifying stage, where one of the classes in the classifier forms a basis to match incoming alarm flood [2]. Support Vector Machines (SVM) and K Nearest Neighbor (KNN) algorithms are used. Alarm floods occurred in a certain time window exceeding the set rate are first identified from the alarms log. Binary series approach is adopted to classify ongoing alarm floods based on these historical alarm floods. The focus is to identify and classify ongoing alarm floods online. Each alarm flood episode is considered as an alarm sequence. This method cannot be applied for alarm rationalization which is an offline method to analyze the alarm history log to help the plant operator. There are two approaches to alarm flood classification which includes similarity based classification and statistical classification of numerical feature vectors.

The papers [3-5] belong to the category of similarity based classification. Similarity based classification is one of the most commonly used approaches. First Nearest Neighbour [3] uses similarity based classification to find how similar a new alarm flood is to alarm floods in the actual data. The author of [4] used an index to measure the dissimilarity also associated with a weight, to compare the list of alarms recorded during an alarm flood to fault templates. This index is compared to Hamming distance, which compares two binary data strings and indicates the number of positions in the bit string at which corresponding bits are different. A resemblance coefficient based on alarm data versus correspondence lag analysis is used to obtain accurate associated variables in [5]. Correlation analysis is done on a combination of alarm and processed data, where Granger causality analysis approach is used to analyze processed data. The author of [6] proposed a representation of alarm coactivation maps to classify alarm floods online based on past episodes. The classification performed by alarm coactivation maps are tested using SVM and KNN classifiers.

Statistical classification of numerical feature vectors focuses on transforming alarm floods into numerical feature vectors [2]. These can be classified using methods similar to those which classify faults in continuous process data. Several graphical and visualization tools were also used for analyzing the performance of alarm systems. These tools help in the assessment of alarm systems, detection of nuisance alarms, detection of correlated alarms, alarm flood analysis, etc. [7]. A High Density Alarm Plot (HDAP) helps in envisaging massive amounts of alarm data for a particular time period. With the help of a colour map, it displays alarm counts for alarm tags whose performance is not so good and provides a very general picture of alarm data without getting into the nuances of each alarm. Alarm Similarity Colour Maps (ASCM) can be helpful in the detection of correlated alarms where the vertical axis and the horizontal axis of this plot indicate alarm tags. The diagonal shows how similar each alarm signal is itself [7]. In [8], Gaussian kernel method is used to transform alarm data which is in binary form to false steady time series data to reduce the effect of abnormal alarms. Considering the time delay between alarm variables, a correlation colour map is used to visualize the correlation symmetric matrix. Here, the variables are arranged in a codified order and highly correlated variables are given the same colour called clusters.

Sequence mining methods are used to analyze alarm data that have been structured as sequences. The different methods include String metrics, sequence alignment, Frequent pattern mining and Association rule mining. Pattern matching algorithms include sequence template extraction and use sequence alignment methods such as Needleman & Wunch algorithm, Modified time distance and weight vectors, Modified similarity scores, Clustering algorithm like Agglomerative Hierarchical Clustering and Dynamic Time Warping. Density Based Spatial Clustering of applications with Noise algorithm with four distance measures, namely Jaccard distance, Levenshtein distance, Euclidean distance and distance based on frequency of alarms. Fast sequence alignment with BLAST, MAA. The pattern matching algorithms in the existing work help in matching an alarm flood sequence with a dataset full of patterns and are checked for similarity or event distance and alarm similarity is calculated between pair of alarm floods. Validation of similarity scores is done using clustering algorithms. Alignment algorithms help in aligning two alarm flood sequences using various strategies like seed and extend, set based pre-matching and backtracking. These methods are useful for the identification or prediction of alarm floods by sequence alignment or sequence matching methods of alarm sequence identified from alarm floods. The alarm flood identification from the alarm log is based on a set time window in which the alarm rates exceed a set rate.

As the alarm sequence identification is specific to alarm floods only, this method cannot be applied for alarm rationalization which would need the alarm sequence identification from the complete alarm log. Also analyzing the complete alarm log would require methods to identify alarm sequences within the complete alarm log.

String metric methods use distance as a metric to find the similarity between two alarm floods. In [9], alarm data analysis involves using Jaccard distance to find the similarity between two alarm floods by calculating the event distance between each pair of alarm floods. There are many types of distance measures which focus on the way alarms appear or frequency of successive alarms and Levenshtein distance which focus on the order of alarms [10]. String metrics has dual applicability here. It can be used for alarm flood template extraction and classification [4].

Sequence alignment focuses on aligning one sequence with the other and then performing a pattern match of the two sequences. The author of [9] proposed dynamic time warping which is used to calculate the distance between time series having different lengths. In [11], the authors used a fast sequence alignment approach to improve the efficiency of the existing algorithms. The basic local alignment search tool (BLAST) is a fast analytical algorithm proposed by Altschul et al. Seed and extend strategy, Scoring based on priority and tolerance based on uncertainty of time are the extensions to BLAST. The author of [12] proposed a modified Smith Waterman algorithm and in [13], MAA (Match-based Accelerated Alignment) was used conducting an alarm match analysis. The main idea behind MAA is that if two sequences resemble each other, then their alignment will be composed of matches.

Frequent Pattern Mining (FPM) methods involve finding the most recurrent patterns in the available sequences of alarms. Algorithms used are Fuzzy association rule mining, Closed Association Rule Mining (CHARM), a combination of context segmentation and frequent itemset mining, Modified PrefixSpan algorithm. Data mining techniques are applied to find frequent pattern of alarms from itemsets. The itemsets are identified from the alarm log as: (i) Each alarm flood defined as a set of alarms in a time window when the alarm rate exceeds a certain preset rate is considered as an itemset; (ii) Correlated alarm sequences are found using the weighted fuzzy association rule mining algorithm; (iii) Return point and activation point strategies are used along with a pre-defined alarm window to identify the itemsets. Frequent pattern of alarms identified from itemsets consisting alarm floods help in identification of alarm floods in an online system. These do not help in alarm rationalization for the complete alarm system. Frequent sequence patterns identified from alarm log using weighted fuzzy association rule or the activation or return point strategy are dependent on the definition of the alarm time window or sliding time window used in this approach. The proposed work provides a method which identifies the frequent sequences in a complete alarm log and not only during alarm floods. Also the alarm sequences (itemsets) are identified from the alarm log in a definitive way based on the alarm start time and Return to Normal for each alarm point in the alarm log. This does away with the need of defining window parameters which would have an effect on the sequences identified. The authors of [14] proposed a frequency based pattern recognition algorithm where the rate of occurrence for the classification of a notification sequence into important and unimportant sequences was used. The author of [15] used CHARM (Closed Association Rule Mining) to find frequent alarm patterns in alarm floods, mine closed alarm patterns and determine characteristic alarm patterns. Frequent Pattern Mining methods can be applied on alarm suppression too.

Association rule (AR) mining includes finding temporal dependencies. In [16], the authors proposed an algorithm to find related dependencies among alarms in alarm logs. Statistical approaches form the basis for determining temporal dependencies between two alarms which may occur several times in an alarm log. In [17], first a sliding time window segments time series data into continuous sub sequences, then a method similar to Apriori method finds frequent itemsets and then extracts association rules from this set. Weighted association rule mining is used to discover correlated alarm sequence. There are several time series analysis methods where alarm data are represented as time series for analysis purposes. These can be further divided into Correlation analysis and Causal inference where correlation analysis focuses on alarm similarity and causal inference focuses on deriving causal relationships between variables in alarm data. In [19], the authors proposed an algorithm to find the most appropriate alignment of many alarm flood sequences. Techniques like Back Tracking, calculation of similarity scoring function, weight vectors and dynamic programming are used. The algorithm proposed in this paper helps in finding a common pattern and further helps in alarm flood analysis. Algorithm complexity is one of the main disadvantages. In [20], the main aim was to find dependencies between alarms by taking into account temporal context and time interval between alarms of different types and using a data mining technique to discover patterns in these alarms based on time intervals. In [21], modified PrefixSpan algorithm was used to find patterns in alarm floods only. Each alarm flood event is taken as an itemset and frequent sequence among these itemsets is identified. A clustering method known as Dynamic Time Warping acts as a

comparison to PrefixSpan algorithm and the work concludes that PrefixSpan works more efficiently.

Visualization of alarm floods algorithms uses Graphical Visualization tools like Alarm Similarity Colour Map (ASCM), High Density Alarm Plot (HDAP) and Gaussian Kernel method. These tools help in finding related or nuisance alarms. They help in analyzing the behavior, studying and monitoring the performance of industrial systems. Gaussian kernel method uses original binary data to generate false continuous time series data. Clustered Correlation colour map represents correlated variables. These methods help in visual representation and analysis of the alarm data and the results are manually obtained. The proposed method uses techniques like frequent sequence mining which analyzes the alarm data log and identifies the frequent sequences in them.

The proposed work is achieving the objective in a different manner from the existing work, i.e., (i) Utilising appropriate data mining algorithms for Frequent sequence mining in place of complex mathematical or statistical or visual methods; (ii) Identifying frequent alarm sequences from the entire alarm data available rather than frequent sequences only from alarm floods; (iii) Representing the alarm data which is Temporal data into itemsets in a definitive way in place of assuming an alarm time window or sliding time window or weighted fuzzy association methods; (iv) The method is scalable to larger size of alarm dataset and implementable on a Personal computer.

III. ALARM PATTERN RECOGNITION USING DATA MINING

One of the most important facilities available in control systems is an Alarm Management system with the Human Machine Interface. Alarm systems are commonly used to monitor process measurement, equipment status, and safety conditions. Fig. 2 is a typical Alarm system depicting the components/ flow from the process to the plant Operators.

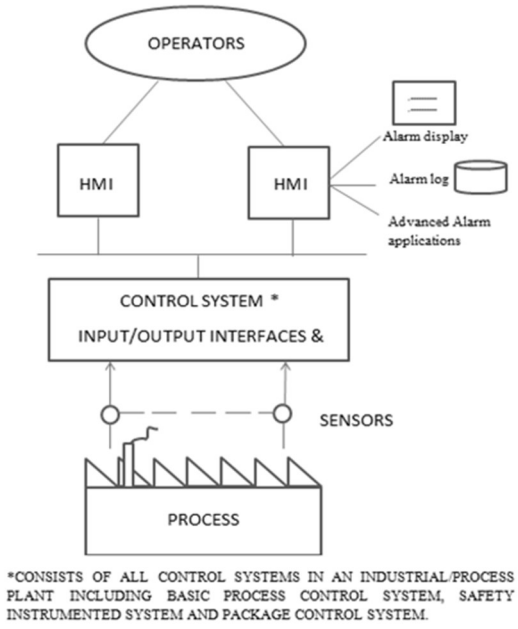

Figure 2. Overview of an Alarm System

The components include the sensors which measure the process parameter, the input/output interfaces which convert the sensor data to digital data, processors which process the data for control and monitoring tasks and the Human Machine Interface (HMI) which presents the Operator with process monitoring/ control data thereby providing a window to the process plant. The alarm screen provided by the alarm management system is one of the displays/ facilities available in the HMI for the operator to monitor the process plant.

The Alarm log stores all the historical alarm and event data from the process plant. Mining of Temporal data such as the alarm and event log is to extract large amount of potentially useful information from data that is represented in the form of a continuous sequence. The process of frequent pattern generation by mining temporal data consists of these steps as shown in Fig. 3:

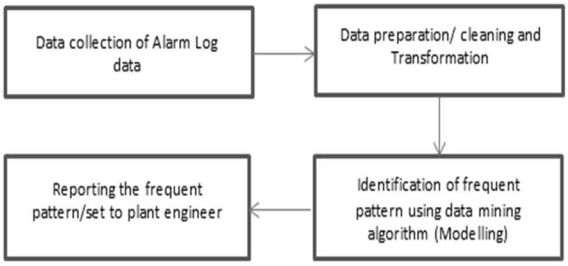

Figure 3. Design Overview

- **Collection of Alarm data**: the original file which is an Alarm and Event log file has variables like date and timestamp, alarm code, Serial Number, source device, Tag Number and description. The alarm log is available in the Control System installed in the process plant and is in the form of a text file or CSV or Excel file. In the present work, we require only three variables, i.e., Date and timestamp, Tag Number and Description.
- **Data preparation**: The Alarm and Event(A&E) log data is then prepared to represent a database with variables such as state, starting time (st) and ending time (et). Here state means the tag number with alarm/event detail, st means the time represented in seconds at which the alarm/event occurred and et means the time represented in seconds at which the alarm/event returned to normal. In the data pre-processing stage, the state would be derived from the Tag number and description data, the start time and end time would be derived from the date and time stamp data from the available Alarm & Event log data.

  Classical data mining techniques to discover frequent sequences need the data to be represented as itemsets. The algorithm then looks for frequent patterns among those itemsets. In an A&E log, the data is of a continuous sequence and hence the basic data is not in the form of itemsets. To represent the A&E log data as itemsets, the definition of frequent patterns in Alarm and Event data is used.

  The frequent patterns in Alarm and Event log data can be defined as -- for a particular process alarm, the subsequent alarm and events that occur frequently with the occurrence of the particular alarm every time with a high support is defined as a frequent sequence of alarms. The temporal data with alarm status, st and et will allow the data to be represented as itemsets

with each itemset consisting of a process alarm (state) and all subsequent alarms and events (states) that occur within the start time and end time of that alarm (state).

So, with a database of 'n' number of process alarms (state) with st and et can be represented as 'n' itemsets. These itemsets can then be subjected to frequent pattern mining algorithms.

- **Identification of frequent pattern using data mining algorithm (Modelling):**.There are different frequent sequential pattern mining algorithms available such as GSP, SPADE, Prefix-SPAN, Bi-Directional Extension, etc. In this project we use the Prefix-SPAN and BIDE algorithm. PrefixSpan focuses on prefix projection in sequential pattern mining. It was proposed by Jian Pei et al. and it uses the horizontal format database. It mines the complete set of patterns without candidate sequence generation. The advantage of prefix projection is that it reduces the size of the projected database so that it leads to efficient processing. In the present work, the PrefixSpan algorithm identifies all possible frequent sequences from the dataset. To identify the frequent sequences/frequent pattern alarms, all the reported subsets of a reported frequent sequence have to be ignored. The longest sequence of the pattern eliminating all its subsets identified by the algorithm is considered as the frequent pattern sequence and reported. The reason for choosing PrefixSpan is its ability to identify frequent sequences even in large size of datasets without exponential increase in the execution time and memory requirements.

  Another sequence mining algorithm found appropriate for the present objective is BIDE (Bi-Directional Extension) which is useful for mining of closed sequential patterns. BIDE proposed by Wang and Han, does not require candidate maintenance for mining frequent closed patterns. BIDE uses BackScan pruning method and Scan Skip optimization technique to prune the search space better than any other closed pattern mining algorithm. It also consumes less memory when compared to other closed pattern mining algorithms. A closed sequence pattern is one which does not have any super sequence among the frequent sequences with the same support as itself. This means that the reported sequential patterns will not have any of its sub sequences having the same support reported. The reported sequential patterns with the BIDE algorithm are more compact (fewer) therefore and without any loss of details of the frequent sequences. The BIDE algorithm also uses a depth first approach. Instead the principle of the forward scan and backward scan incorporated in the algorithm enables the closure of the identified patterns and pruning of the search space.
- **Reporting** The frequent pattern/set of alarms identified by the mining algorithm is reported along with the support for that sequence within the input dataset. This can be used by the plant engineer to optimise or rationalise the alarm configuration to make the alarm system more effective to the Plant operation and reduce the flooding of alarms in the Control system HMI.

A. IDENTIFICATION OF FREQUENT PATTERN USING PREFIX SPAN AND BIDE

The implementation is done using python on Pycharm IDEand the same was also adapted to the Jupyter notebook.

The program is given an Input as an Excel file prepared from an Alarm and Event Log from a control system consisting of variables such as state, start time (st) and end time (et). The start time (st) and end time (et) is represented in seconds to facilitate the easy processing of the dataset from the Input data. The Date and time stamp of the first alarm in the alarm and event log file from the control system is taken as reference and all other alarm end event timestamp is taken relative to that, i.e., the first alarm in the input file to the program would have st as 0 and all other st and et is represented in seconds relative to the reference date and time. The algorithm for Identification of frequent pattern using Prefix span and BIDE is given in Algorithm 1.

The Input data which is temporal data is next represented as a dataset of Itemsets. Each Itemset consists of a set of multiple Items – each item being an Alarm (st). For a particular alarm (state) in the input data, the corresponding itemset would have this alarm (state) and all other alarms (states) that have occurred between the st and et of this alarm (state). Hence, the Input data with n number of alarms is represented as a dataset with n Itemsets. Each Itemset is considered as an individual sequence for input to the data mining algorithm. With the occurrence of an alarm (state) multiple times within the Input data, multiple sequences would be present in the prepared data set for the same alarm (state).

The set of sequences for each alarm (state) is then processed by the PrefixSpan algorithm to identify the repetitive sequence of alarms (state). A confidence factor or minimum support of 0.9 is considered for reporting the frequent repetitive sequences. As the intent is to identify frequent sequences, a sequence with a single item is not of interest. Hence, Minimum sequence length is defined and only sequences longer than the set minimum sequence length is considered for reporting. The Prefix span algorithm requires the Maximum pattern length to also be defined. This value is set as a high number so that all the frequent sequences with the maximum possible length are identified from the input dataset and reported. The identified frequent repetitive sequences along with the support, i.e., the number of occurrences of the repetitive sequence in the dataset and the maximum possible support (occurrences) for the shortest sequence in the dataset are all reported and output as an Excel file. The set of sequences for each alarm is also processed by the BIDE algorithm, which takes minimum support, minimum length and maximum pattern length as its parameters. The most frequent closed sequence with its minimum support, count of itemset and length of database are reported and output as an excel file. The reason for choosing BIDE for a comparative view is due to the ability of the algorithm to efficiently mine sequential patterns from large datasets. The expectation of the algorithm is to scale linearly with increase in size of datasets and patterns, i.e., a linear increase in requirements of CPU and memory with increase in dataset size. Also, the benefit of the algorithm is that it mines closed sequential patterns, inherently eliminating the reporting of subsequences of a larger identified sequence with the same support.

To consider the performance of the implemented algorithm and confirm the scalability, the performance parameters on each run of the program are collected. With the program run

with incremental size of the Input data, these factors can be evaluated.

B. PROPOSED ALGORITHM 1:

Input: Temporal data set with set of state (alarm), Starting time (st) and ending time (et).
Output: A set of the longest repetitive sequence with support and maximum support among all sequences.
Read Input dataset as Dataframe
For every state in Input dataset
    Read state, st and et.
    Check et>st
    if et > 0
setst as stRef, et as etRef and state as stateRef
Append stateRef toitemset
For each state in dataset except stateRef
    Read state and st
    ifstRef<st<etRef
    Append state to itemset
Iteration performance optimized with start iterrow on rowRef of stateRef and iterbreak on etRef
        Append itemset to sequence dataset
List Uniquestateat location element1 of all itemset in sequence dataset
For every UniquestateasUniqueref
Check element1 of each itemset in sequence dataset to match Uniqueref
Append seq to Sequenceset
    Sequenceset=Sdb
    MinSupport threshold=min_support
    Call PrefixSpan(Sdb, min_support)
For every frequent sequence fs with its support supp
    Check fs >minPatternLength
    fsresult= fs with maxPatternLength
    suppresult= supp of fsresult
    supp_max = max supp among all fs
append(fsresult,suppresult, supp_max) to res_out
Convert res_out to dataframe and output to excel file

IV. RESULTS AND DISCUSSION

This section discusses the results obtained. The program was run on two different sets of data with incremental size of input data. The alarm log from the control system is taken as CSV or excel files. Each file contains a certain number of alarms. With alarm log taken for a long duration of time, it would consist of multiple files. For the purpose of this work, the scalability of the studied method for application on larger size datasets was also an objective. Therefore, the implementation was studied with increasing size of the input data file with the smallest dataset being data from a single alarm log file (No: of states = 1476) and the largest dataset being data from all the alarm log files (No: of states = 22779).

The data taken from the alarm log would be as YYYY/MM/DD, HH:MM:SS, Tag No., Alarm description. During data preparation from the alarm log, this data is represented as=Tag No-AL, Start time, End time where Tag No-AL is the combination of the Tag No and alarm description and it is called as the **State.** The start time is based on the date and timestamp of the alarm occurrence represented in seconds with reference to the date and timestamp of the first alarm in the alarm log and the end time is also represented in seconds based on the date and timestamp of the alarm returning to normal. The frequent sequence mining algorithm considers the input data in the following representation: **Itemset:** {Tag A–AL,……..,Tag Z–AL}. Here **Itemset** is the list of **items** which in this case is a list of alarms (State) which occur during the time a particular alarm X occurs and returns to normal. The itemsets are derived from the data prepared from the alarm log based upon the start time of all alarms (States) in the alarm log and the start time and end time of the particular alarm X.

The analysis is done with respect to execution time for the following variables. (1) Itemset count -The itemset is also considered as a sequence in the proposed work. The count of all itemsets in the input data file for discovering frequent sequences is represented as the **Itemset count.** The total of all items within these itemsets is represented as the **Item count** (No: of items). (2) Number of states – The total number of States derived from the alarm log is represented as Number of states. (3) Number of frequent sequences. (4) Dataset execution time (Dataset execution time is the time taken to represent temporal data as itemsets). (5) Total program time -The time taken by the program to discover the frequent sequences in an input data file is represented as the **Program execution time** (total program time).

The minimum support was kept at 0.9. Fig. 4 is a graph of Itemset count vs Dataset execution time for PrefixSpan algorithm and BIDE algorithm, where it is observed that as the number of itemsets increases, the dataset execution time also increases linearly.

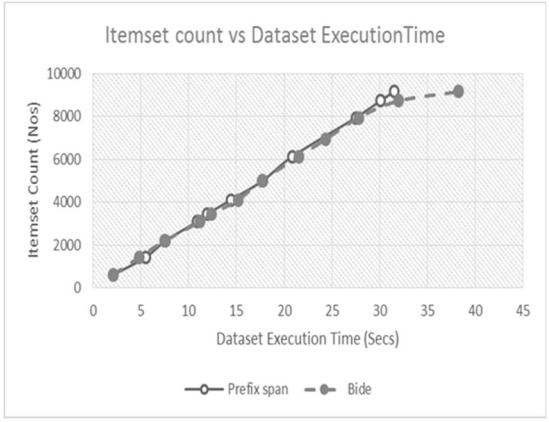

Figure 4. Itemset Count (no:s) vs Dataset execution time (secs) for PrefixSpan and BIDE algorithm

Fig. 5 shows the relationship between Number of states (alarms) in the alarm log file and Total program time for PrefixSpan algorithm and BIDE algorithm.

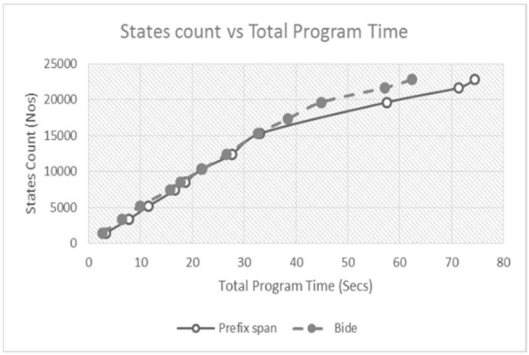

Figure 5. No: of states (no:s) vs Total program time (secs) for PrefixSpan and BIDE algorithm

Fig. 6 is a graph of Number of items vs Total program time for PrefixSpan algorithm and BIDE algorithm. It is seen that as the number of items increases, the total program time also increases linearly.

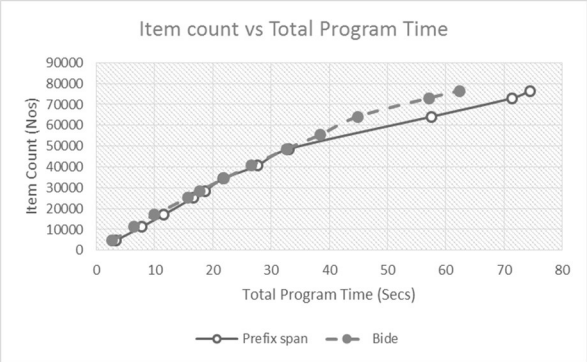

Figure 6. No: of items (no:s) vs Total program time (secs) for PrefixSpan and BIDE algorithm.

Fig. 7 is a graph of Itemset count vs Total program time for PrefixSpan algorithm and BIDE algorithm. The graph is observed to be linear in nature.

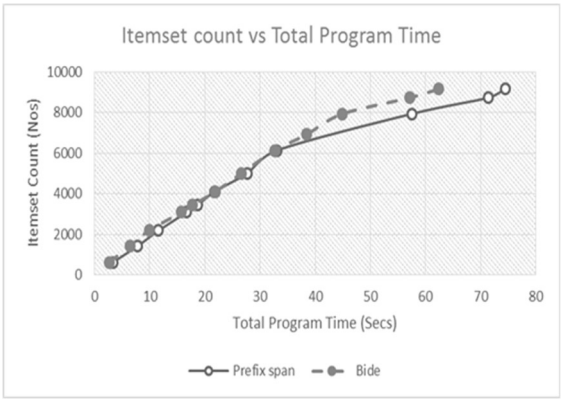

Figure 7. Itemset count (no:s) vs Total program time (secs) for PrefixSpan and BIDE algorithm

Fig. 8 shows the comparison of BIDE and PrefixSpan algorithm based on total program time and the graph is close to linear in nature. The graphs for both algorithms are observed to be close to linear in nature but the BIDE algorithm is observed to be faster than the PrefixSpan algorithm.

The drawback observed in the implementation is that PrefixSpan algorithm identifies all alarm sequences which includes subsequences of a larger sequence having the same support. If the subsequences were not to be reported, the algorithm would have worked much faster which would help when the database size is large. The BIDE algorithm identifies all the closed sequential patterns and reports all the sequences which meet the minimum support criteria. This could mean that many frequent sequential patterns are reported and not the longest frequent sequence pattern only – which could at times lead to loss of focus on the important frequent sequence pattern in the result output. For the present objective, it could be the best way if only the longest closed frequent sequence pattern with its support and the sequence pattern with the highest support both meet the criteria of the minimum support specified.

The results show that the graphs are close to linear in nature. Based on the performance of the chosen algorithm with the varying sizes of the datasets, it is seen that both PrefixSpan and BIDE algorithms scale up linearly with the size of the dataset and the BIDE algorithm being faster than the PrefixSpan algorithm. The linear performance establishes the fact that these algorithms can be utilized even on larger size datasets. (If the algorithms scaled exponentially then they could not have been utilized on larger size datasets).

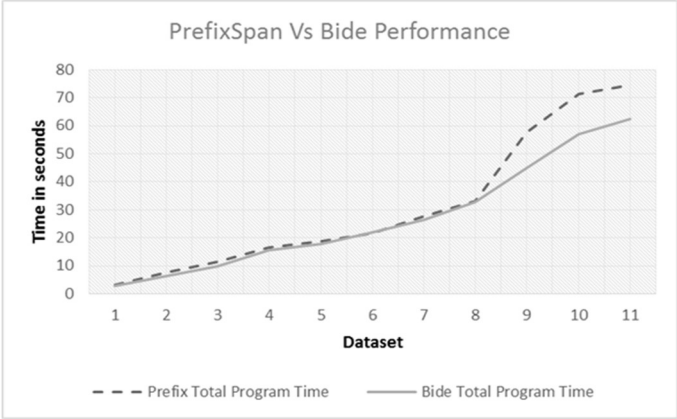

Figure 8. Comparison of Prefixspan and BIDE based on total program time

Table 1 is a comparison of the method and results between existing work in [21] and the proposed work. It is seen that the frequent sequences can be identified from the complete alarm log data and not just the alarm floods. The frequent sequences thus discovered would indicate the chattering alarms, duplicate alarms and sequence patterns in all alarm episodes including alarm floods. This would facilitate the results of the output to be used for alarm rationalization and also for further analysis of the alarm system and identification of alarm floods.

Table 1. Comparison of the method and results between existing work [21] and the proposed work

Parameter	Existing work	Proposed work
No: of alarms in alarm log	18,000	A dataset of similar size is selected for comparison – about 22,000.
Chattering alarm Filtering definition	An alarm repeating more than once in a window of 100 seconds is identified and eliminated from the dataset.	There is no filtering to be set. Chattering alarms can be found from the result of sequence mining algorithm.
Number of alarms in data set used for evaluating sequence patterns	440	Same as dataset selected – about 22,000
Identifying alarm sequence itemsets	Each alarm flood is identified as an itemset. (>10 alarms in 10 minutes at (the start of the flood) to alarm rate zero).	Alarm sequence itemset identified for each alarm point in alarm log based on alarm start time and Return to Normal time.
Number of itemsets for analysis	9 itemsets	9,163 itemsets
Minimum support used for pattern extraction	2	Alarms in frequent sequence should appear 8 out of 10 times.
Minimum pattern length set	25	2
Algorithm output	Sequence pattern in alarm flood	Chattering alarms, duplicate alarms and sequence patterns in all alarm episodes including alarm floods.

V. CONCLUSION AND FUTURE WORK

This paper highlights the tools and techniques applicable to Online alarm flood classification, alarm flood pattern/sequence recognition and visualization tools along with the advantages and disadvantages of each of these methods. The steps followed in frequent pattern generation, the PrefixSpan and BIDE algorithms and their implementation in the project, are also discussed. It is seen that commonly used frequent sequence mining algorithms can also be effectively utilized for identifying frequent sequence patterns in an Alarm/Event database which is temporal data. This approach also eliminates the requirement of any complex algorithms for processing of the input data prior to frequent sequence mining by the data mining algorithm. Also, the methodology used for representation of the temporal data (alarm data) as itemsets does not utilize any assumptions or approximations in the approach thereby making the result accurate and definitive. Also it is demonstrated that the resource requirements for the frequent pattern recognition could be easily met by normally available computers without the need for high end computation platforms. The computation time required was found to increase linearly with the increasing size of dataset.

Further development in code in terms of processing input data from alarm log file and optimizing the program run by considering the previously identified sequences as initial sequence set, can be done, so that larger datasets can be run in shorter time. As future work, better algorithms can also be identified, so that only the longest sequences or maximum support sequences can be identified.

References

[1] ISA., *Management of Alarm Systems for the Process Industries*, 2009, 75 p.

[2] M. Lucke, M. Chioua, C. Grimholt, M. Hollender, N. F. Thornhill, "Advances in alarm data analysis with a practical application to online alarm flood classification," *Journal of Process Control*, vol. 79, pp. 56-71, 2019. https://doi.org/10.1016/j.jprocont.2019.04.010.

[3] S. Charbonnier, N. Bouchair, P. Gayet, "Fault template extraction to assist operators during industrial alarm floods," *Engineering Applications Artificial Intelligence*, vol. 50, pp. 32-44, 2016. https://doi.org/10.1016/j.engappai.2015.12.007.

[4] S. Charbonnier, N. Bouchair, P. Gayet, "A weighted dissimilarity index to isolate faults during alarm floods," *Control Engineering. Practice*, vol. 45, pp. 110-122, 2015. https://doi.org/10.1016/j.conengprac.2015.09.004.

[5] S. Lai, F. Yang, T. Chen, "Online pattern matching and prediction of incoming alarm floods," *Journal of Process Control*, vol. 56, pp. 69-78, 2017. https://doi.org/10.1016/j.jprocont.2017.01.003.

[6] M. Lucke, M. Chioua, C. Grimholt, M. Hollender, N. F. Thornhill, "Online alarm flood Classification using alarm coactivations," *International Federation of Automatic Control (IFAC) Papers Online*, vol. 51, issue 18, pp. 345–350, 2018. https://doi.org/10.1016/j.ifacol.2018.09.324.

[7] W. Hu, M. S. Afzal, G. Brandt, E. Lau, T. Chen, S. L. Shah, "An application of advanced alarm management tools to an oil sand extraction plant," *International Federation of Automatic Control (IFAC) Papers Online*, vol. 48, issue 8, pp. 641–646, 2015. https://doi.org/10.1016/j.ifacol.2015.09.040.

[8] F. Yang, S. L. Shah, D. Xiao, T. Chen, "Improved correlation analysis and visualization of industrial alarm data," *Proceedings of the 18th World Congress of the International Federation of Automatic Control,* Milano, Italy, 2011, pp. 12898-12903. https://doi.org/10.3182/20110828-6-IT-1002.03193.

[9] K. Ahmed, I. Izadi, T. Chen, D. Joe, and T. Burton, "Similarity analysis of industrial alarm flood data," *IEEE Transactions on Automation Science and Engineering,* vol. 10, no. 2, pp. 452-457, 2013. https://doi.org/10.1109/TASE.2012.2230627.

[10] M. Fullen, P. Schuller, O. Niggerman, "Validation of similarity measures for industrial alarm flood analysis?" In: Niggemann, O., Schüller, P. (eds) *IMPROVE – Innovative Modelling Approaches for Production Systems to Raise Validatable Efficiency. Technologien für die intelligente Automation*, vol. 8, 2018, pp. 93-109. Springer Vieweg, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-57805-6_6.

[11] W. Hu, J. Wang, T. Chen, "Fast sequence alignment for comparing industrial alarm floods," *International Federation of Automatic Control (IFAC) Papers Online*, vol. 48, issue 8, pp. 647–652, 2015. https://doi.org/10.1016/j.ifacol.2015.09.041.

[12] W. Hu, J. Wang, T. Chen, "A local alignment approach to similarity analysis of industrial alarm flood sequences," *Control Engineering. Practice*, vol. 55, pp. 13-25, 2016. https://doi.org/10.1016/j.conengprac.2016.05.021.

[13] C. Guo, W. Hu, S. Lai, F. Yeng, T. Chen, "An accelerated alignment method for analysing time sequences of industrial alarm floods," *Journal of Process Control*, vol. 57, pp. 102-115, 2017. https://doi.org/10.1016/j.jprocont.2017.06.019.

[14] B. Vogel-Heuser, D. Schutz, J. Folmer, "Criteria based alarm flood pattern recognition using historical data from automated production system (aPS)," *Mechatronics*, vol. 31, pp. 89-100, 2015. https://doi.org/10.1016/j.mechatronics.2015.02.004.

[15] W. Hu, T. Chen, S. L. Shah, "Detection of frequent alarm patterns in industrial alarm floods using itemset mining methods," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 9, pp. 7290-7300, 2018. https://doi.org/10.1109/TIE.2018.2795573.

[16] J. Folmer, F. Schuricht and B. Vogel-Heuser, "Detection of temporal dependencies in alarm time series of industrial plants," *Proceedings of the 19th World Congress of the International Federation of Automatic Control*, Cape Town, South Africa, 2014, pp. 1802-1807. https://doi.org/10.3182/20140824-6-ZA-1003.01897.

[17] J. Wang, H. Li, J. Huang, C. Su, "Association rules mining based analysis of consequential alarm sequences in chemical processes," *Journal of Loss Prevention in Process Industries*, vol. 41, pp. 178-185, 2016. https://doi.org/10.1016/j.jlp.2016.03.022.

[18] S. R. Kondaveeti, I. Izadi, S. L. Shah, T. Black. "Graphical representation of industrial alarm data," IFAC Proceedings Volumes, vol. 43, issue 13, pp. 181-186, 2010. https://doi.org/10.3182/20100831-4-FR-2021.00033.

[19] S. Lai, T. Chen, "Methodology and application of pattern mining in multiple alarm flood sequences," *International Federation of Automatic Control (IFAC) Papers online*, vol. 48, issue 8, pp. 657–662, 2015. https://doi.org/10.1016/j.ifacol.2015.09.043.

[20] S. Kordic, C. P. Lam, J. Xiao, H. Li, "Patterns relevant to temporal data-context of an alarm of interest," in Dynamic and Advanced Data Mining for Progressing Technological Development: Innovations and Systemic Approaches, 2010, pp. 18-39. https://doi.org/10.4018/978-1-60566-908-3.ch002.

[21] T. Niyazmand, I. Izadi, "Pattern mining in alarm flood sequences using a modified PrefixSpan algorithm," ISA Transactions, vol. 90, pp. 287-293, 2019. https://doi.org/10.1016/j.isatra.2018.12.050.

***CHETANA BELAVADI** received her B.E degree in Computer Science and Engineering from B.M.S Institute of Technology (VTU), Bangalore, Karnataka, India in 2018 and MTech degree in Computer Science and Engineering from Ramaiah Institute of Technology, Bangalore, Karnataka, India, in 2020. She is working as an application analyst in an IT company. Her interests include Data mining and Machine Learning.*

VANDANA SUDHAKAR SARDAR *received her B.E degree from Pune University, India in the year 2000 and M.E degree from Pune university, India in the year 2008. She is currently working as Assistant professor in Ramaiah Institute of Technology, Bangalore, Karnataka, India. She has published several research papers in National and International conferences. She is a Life member of Indian Society for Technical Education (ISTE). Her area of interest includes Information retrieval and Machine Learning.*

DR. SHILPA SHASHIKANT CHAUDHARI *is currently working as an Associate professor in Department of Computer Science and Engineering, Ramaiah Institute of Technology, Bangalore, India. She has completed her Ph.D. from Visvesvaraya Technological University, Belgaum at REVA ITM, Bangalore. She has been a technology educator and corporate trainer since 1999. Her research contribution on cloud/network security and wireless networks exists in various national and international indexed journals and conferences. She is a technical reviewer for a good indexed international journal.*