Alarm flood reduction using multiple data sources

Vicent Rodrigo Marco

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
ISRN LUTFD2/TFRT--5961--SE
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Media-Tryck
Lund 2014

Abstract

The introduction of distributed control systems in the process industry has increased the number of alarms per operator exponentially. Modern plants present a high level of interconnectivity due to steam recirculation, heat integration and the complex control systems installed in the plant. When there is a disturbance in the plant it spreads through its material, energy and information connections affecting the process variables on the path. The alarms associated to these process variables are triggered. The alarm messages may overload the operator in the control room, who will not be able to properly investigate each one of these alarms. This undesired situation is called an “alarm flood”. In such situations the operator might not be able to keep the plant within safe operation. The aim of this thesis is to reduce alarm flood periods in process plants. Consequential alarms coming from the same process abnormality are isolated and a causal alarm suggestion is given. The causal alarm in an alarm flood is the alarm associated to the asset originating the disturbance that caused the flood. Multiple information sources are used: an alarm log containing all past alarms messages, process data and a topology model of the plant. The alarm flood reduction is achieved with a combination of alarm log analysis, process data root-cause analysis and connectivity analysis. The research findings are implemented in a software tool that guides the user through the different steps of the method. Finally the applicability of the method is proved with an industrial case study.

Acknowledgements

I would like to use this oportunity to extend my sincere thanks to all the people without whom this thesis would not have been possible.

I wish to thank, first and foremost, my supervisor Dr Moncef Chioua (researcher in ABB Corporate Research Center in Ladenburg, Germany) for his patient guidance, encouragement and his critical feed-back. He helped me to shape my thoughts in the right direction, not only on this work but also for my future as an engineer. I consider myself extremely lucky to have such a talented supervisor that was always available to respond to my questions and queries.

Besides my advisor, I would like to thank Dr Martin Hollender for his encouragement and insightful comments.

I dedicate this thesis to my parents, Salva and Rosa, and my brother Carles for their love, advices and unconditional support throughout my life.

Contents

1

Introduction

This thesis presents the findings from the research conducted into establishing innovative ways to manipulate process plant data to facilitate the supervision and monitoring of process plants. An off-line method to reduce alarm flooding in alarm monitoring systems is proposed and a software tool that proves practical applicability of the research findings is implemented.

1.1 Alarm Systems

The profit in the process industry is highly related to the plant operation. Over the last decades advanced operation and control approaches have been developed in order to optimally operate plants. This is done by keeping process variables at a specific value [Christofides et al., 2007]. A deviation from the optimal operation point is usually translated into an economic loss or endangering the environment or the safety of the personnel. The operators working in the control room are in charge of keeping the plant operating at this point.

An alarm system is the main element that interfaces in modern plants with the operator to the plants. It is a crucial element since it monitors the plant operation and alerts the operator when some undesired state that requires his assessment or action is reached. Its main objectives are: to assist the operator to correct potentially dangerous situations before the emergency shutdown system is triggered, to avoid financial loss by identifying deviations from the optimal operating conditions and to help the operator to better understand the process conditions that gave rise to the upset. The primary function of an alarm system is defined by the EEMUA as follows:

The purpose of an alarm system is to direct the operator's attention towards plant conditions requiring timely assessment or action [EEMUA-191, 2007].

1.1.1 Alarms in Process Industry

An alarm is a signal sent to the operators in order to draw their attention during an abnormal situation in the plant. An alarm occurrence comes usually together with a sound, flashing light and an indicator. In addition, alarms usually present a message with some information about the problem.

Every alarm presented to the operator should be useful and relevant to the operator [EEMUA-191, 2007]. Ideally, for each abnormal situation just one alarm is raised. However, operators usually receive a large amount of alarms in practice. Most of these alarms are either false alarms, i.e. alarms that alert of an abnormal situation when there is none, or nuisance alarms, i.e. alarms that are redundant, since other alarms have already informed of the abnormality.

Traditionally an alarm in a process plant is associated to a process variable going out of its normal range (see Figure 1.1).

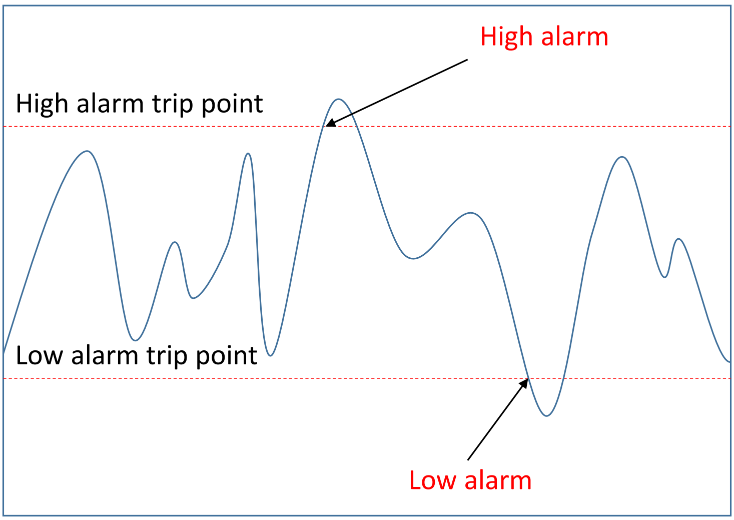


Figure 1.1 Signal alarm

These type of alarms are called absolute alarms. However, one should bear in mind that many other alarm detection mechanisms are used in practice, and not all alarms are strictly associated to a process signal. Some of these mechanisms are listed below (for more information the reader is referred to [EEMUA-191, 2007]):

- bit-pattern alarms
- discrepancy alarms
- calculated alarms
- recipe-of-change alarms
- statistical alarms
- adaptive alarms
- ...

1.1.2 Alarm Floods, a source of plant incidents

In the past, alarms systems consisted of lamps and horns individually connected to the process [*Alarm Management: A Comprehensive Guide, Second Edition*]. Since setting an alarm with this conditions was costly and the space in the control room was limited, every alarm was well thought through.

With the digital revolution, the introduction of Distributed Control Systems (DCS) in the industry together with the reduction of the price of computer hardware, the situation changed. An alarm nowadays is listed in a scrollable table or on graphics, therefore there is no space limitation. Besides, adding a new alarm to the alarm system is as easy as writing some code lines or clicking on the proper options. There is no need of connection planning, no need of wiring, just typing. The costs are thus practically non-existent. As a result, the number of configured alarms per operator has increased exponentially in the last decades due to an abuse of the use of alarms [*Alarm Management: A Comprehensive Guide, Second Edition*].

On the other hand, the increasing number of process variables present in monitoring systems as a result of the technological evolution in the past decades has brought many benefits: higher product quality, improved emissions control, and higher productivity among others. New control algorithms have been developed to increase the efficiency of plants. Control algorithms have become more complex and more process variables are needed in the control loops. In addition to this, due to the progress in sensor technology, a wider variety of process variables are now accessible. However, each process variable brings along at least four alarms which overloads the alarm system and makes the supervision of the process more complex.

Furthermore, modern plants are highly interconnected due to steam recirculation, heat integration and complex control approaches to increase the efficiency of the plant. A single disturbance spreads fast through the plant due to these material, energy and information connections triggering the alarm messages connected to process signals in the path.

All of these elements lead to the regularity of the situation where the operator in the control room is overloaded with a large amount of alarms. The operator cannot take the time to properly analyse each alarm, which may cause critical alarms to be ignored (e.g. acknowledged alarms not being properly investigated and corrected). Missing a critical alarm translates into material losses, endangers the integrity of the plant or even puts human life at risk.

The situation in which the operator is overwhelmed with a high number of alarms is known as alarm flood or alarm shower. Alarm floods have been cited as a significant cause to most industrial incidents investigated by the US Chemical Safety Board (CBS) [Duisting Beebe and Logerot, 2007]. The definition of an alarm flood is given in [ANSI/ISA-18.2, 2009] as:

> "A condition during which the alarm rate is greater than the operator can effectively manage (e.g., more than 10 alarms per 10 minutes)."

In order to illustrate the relevance of the problem, some real incidents that gave motivation for this thesis are presented [Bransby and Jenkinson, 1998]:

"On a large petrochemical plant a trip occurred on a compressor. There were two days lost production before the plant was put back on-line (cost around £1M). Also, components in the plant were damaged by the trip, and some months later the plant had to be taken off for an unscheduled outage to repair this damage. This outage cost about £12M, but other work was done in this outage, so not all the cost can be attributed to the trip. Six weeks prior to the compressor trip there had been an alarm on high axial displacement which the operators accepted but did not investigate. Three days prior to the trip there was a second similar alarm which also was not investigated".

"On 24th July 1994 there was a major explosion at the oil refinery at Milford Haven jointly owned by Texaco and Gulf. There was plant damage that cost about £48M to repair. There was also two months lost production from the complete plant and four months lost production from the area that was damaged.[. . .] Alarm system shortcomings were one major contributor to this incident. There was a lightning strike which caused a significant plant upset. For several hours after the lightning strike the operators were heavily loaded with alarms at a rate estimated to be in excess of 1 every 2-3 seconds. During this period several operators failed to identify the buildup of liquid in a knock-out vessel. This eventually overfilled and resulted in the explosion taking place. A number of instrument faults contributed

to the operating confusion.[...] A large number of people (26) sustained minor injuries as a result of the explosion".

After the incident of the oil refinery at Milford Haven, the British authorities demanded the Engineering Equipement & Materials Users' Association (EEMUA) to establish a guide for the design of alarm management systems. Both EEMUA guide and ISA Standard state that an operator should not receive more than six alarms per hour. However, in most plants much higher numbers are encountered. Situations where the operator receives too many alarms are common.

In [Henningsen and Kemmerer, 1995] it is stated that in a typical alarm log, a big part of the unimportant alarms falls in the following categories:

- Repetitive alarms
- Standing alarms
- Consequence alarms

For removing repetitive and standing alarms a large variety of methods is available in the literature, such as: time delays, filtering (median filter, average filter, IIR filter), dead band, difference functions and alarm window functions. Both academia and industry have studied the reduction of these kinds of alarms for years. Intelligent tools for reduction of nuisance alarms have been developed. An example of these tools is the toolbox created at Lund Institute of Technology described by Jonas Ahnlund et al. in [Ahnlund et al., 2003] and [Bergquist et al., 2003]. The toolbox uses a system called LARA (Logical Alarm Reduction Algorithm) that first classifies the alarm signals into predefined groups and later applies the proper alarm handling algorithms according to the signal group it belongs to. These techniques succeed in lowering the number of alarms when the process is running in nominal conditions. Nevertheless, when a fault or abnormality occurs and it propagates through the plant, these methods can get rid of some of the alarms but they cannot solve the problem of suppressing consequence alarms. Filtering consequence alarms is quite complicated and requires analysing causal relationships in the process.

1.2 State of the Art

Recently, both industry and academia have put effort on reducing alarm floods in the process industry. Researchers have focused on grouping consequential alarms, i.e., isolate those alarms caused by the same event.

In [Higuchi et al., 2009], Fumitaka Higuchi et al. applied event correlation analysis to alarm data from an ethylene plant in Japan in order to reduce the amount of alarms. The similarity grade between alarms is determined by computing their cross correlation function. After that, the alarms are grouped using hierarchical clustering. Junya Nishiguchi et al. showed the benefits of the same method with a case study of a chemical site in [Nishiguchi and Takai, 2010]. Once the alarms are clustered according to their similarity, an analysis of the results is performed. The clusters obtained are classified according to their relationships. Consequential alarms, complex operator actions (i.e. sequential operations that could be automatized), unnecessary alarm and causes of upset are some of the possible relationships between events in a certain cluster. However, to be able to calculate the event cross correlation properly, knowledge of the time-delay between the alarms and the operator response is required.

F. Yang et al [Yang et al., 2012] propose a method to group correlated alarms by generating continuous time pseudo data from the original binary alarm record via a Gaussian kernel method. Once the data is transformed from discrete to continuous, the pseudo data is being analysed using statistical approaches (correlation and singular value analysis). Using pseudo data instead of binary alarm data proved to have some advantages: it reduces the influence of missed, false and chattering alarms, the correlation analysis provides information about the direction of the correlation (positive or negative), statistical approaches to analyse the data can be used and information about the delay between alarms is not necessary.

Other authors suggest using sequence pattern matching for grouping consequential alarms (e.g. [Kabir et al., 2013], [Cheng et al., 2013]). The authors of [Kabir et al., 2013] and [Cheng et al., 2013] have noticed that a large portion of the alarm occurrences in an alarm flood is related and follows a specific pattern. If similar alarm floods are identified, it is easier to identify this pattern. The alarms contained in this pattern are likely to be consequential alarms. Several methods for clustering similar alarm floods have been proposed. In [Kabir et al., 2013] Kabir Ahmed et al. present a method for clustering alarm floods according to the patterns of the alarm sequences. In this approach, the time difference among the alarms is disregarded. The similarity between different alarm flood sequences is determined under the assumption that the alarms appear in a cause-consequence manner in the alarm log. If we have a sequence [A, B, C, D] it is expected that B is caused by A, C by B and so forth. However, this assumption may not be true for all cases as the order of the alarms is highly dependent on their alarm limits. For instance, if alarm A has narrow alarm limits while alarm B limits are set wide, even though the disturbance originates from B and propagates to A, the alarm notification is first triggered in A and later in B.

Additionally, Yue Cheng et al [Cheng et al., 2013] pointed out that the order of appearance of the alarms in the sequence should not be taken at face value. The detection delay of an abnormality is a probabilistic variable. It could be that a number of linked alarms trigger practically at the same time and the order in which these alarms activate changes from case to case. Therefore, the exact order of appearance of the alarms is not as important as their proximity in time. The authors of [Cheng et al., 2013] presented an alternative method for pattern matching of alarm floods based on the Smith-Waterman algorithm. In the algorithm the time-stamp of the alarm sequence is taken into account in order to blur the order in the alarm sequence when the alarms happen close in time.

Markus Schleburg et al [Schleburg et al., 2013] suggested an automated method to reduce the alarms presented to the operator by grouping consequential alarms. In order to do so not only the process alarm log are used (as in the previous cited methods), but also a description of the plant topology (*P&ID*) and a set of rules representing process knowledge. In this method, every possible pair of alarms is checked against the rules and the plant topology to see if they are related or not. The rules apply to different properties of the alarms such as the allowed time-frame between alarms, their type (pressure, liquid flow, temperature), their status, type of connection (physical or signal connection) and to which assets are to be found between the locations of the alarms. Grouped alarms are shown clustered in the alarm log. Solely the first and the last alarm in time are displayed. All other alarms within the group are found by clicking on the first alarm. The method developed by Markus Schleburg et al. was proved to be a promising technique to reduce the number of alarms in an alarm log by using process knowledge translated into tunable interrelation rules. However, for the grouping tool to be useful, additional knowledge about the plant is needed and it also requires some time from the operators to set the necessary rules to adapt the method to a specific plant.

On the other hand, in the last years, several contributions from both academia and industry investigated the design of systematic methods that search for fault propagation paths and perform root-cause analysis of plant-wide disturbances. Data-driven methods that detect directionality between measurements have emerged. Time delay analysis captures the causality relationship between process variables using the time delay between them [Fan and Deyun, 2012]. In [Bauer et al., 2007] Margret Bauer et al. demonstrate the effectiveness of a data-driven method called Transfer Entropy that identifies the causality between measurements even in absence of observable time delays between the process variables.

At the same time, the use of qualitative models in the area of fault diagnosis has been an active area. Signed direct graphs (SDG) are a type of qualitative model. They are able to describe causal relationships between process variables. In SDGs, process variables are represented as nodes and the causal influence between them (positive or negative) as arcs [Fan et al., 2010].

Recently, the benefits of combining both quantitative process history methods and qualitative models have been studied. When used independently these methods can give rise to spurious results. The aim is to use data-driven analysis methods to generate hypothesis regarding the possible root-cause, and after that use qualitative models (e.g. process connectivity) to validate the obtained result, i.e., if it exists an actual physical or logical connection between the disturbed signals. In this way, the number of spurious results is reduced. In [Thambirajah et al., 2009], Jagatheeswaran Thambirajah et al. show the advantages of this combination with an industrial case study. A connectivity matrix is derived from a CAEX description of the process to explore the topology of the plant. The results obtained from the analysis were validated by process experts.

In Summary, automatized methods for grouping consequential alarms are proposed in the literature. These methods rely on alarm sequence analysis or on a combination of process connectivity and case base reasoning. On the other hand, causal root cause analysis of disturbance propagation enable to determine the propagation path of a disturbance within a process using process data and process connectivity information.

1.3 Scope

Within the last years, advanced alarm management systems and precedence (root-cause) analysis for process plants have received an increasing attention from both industrial and academic communities. In the alarm flood reduction area, alarm records have been the traditional information source to identify related alarms. However, recent studies have shown the benefits of combining alarm logs with plant connectivity information to identify alarms caused by the same event. On the other hand, methods for root-cause analysis using both data-driven analysis and topology (plant connectivity) analysis have proved to be successful in finding the root-cause of a plant upset.

The contribution of this thesis is to show the advantages of using a combination of alarm, process and topology data to not only identify related alarms, but also isolate the causal-alarm (alarm closest to the root) performing a precedence analysis. To the authors' best knowledge a method to identify the causal alarm within an alarm flood using data-driven and topology analysis is still not available.

The goal of this study is to lower plant incidents by developing a method for the reduction of alarm flood periods using multiple available information sources. In addition, the research findings are implemented in a software application that will basically guide the user through the different steps of the method. However, extra information that helps the user to better understand the problem analysed or the method is added.

2

Pattern analysis of Alarm sequences

In the past years, statistical pattern recognition has been found to be successful in numerous applications such as genomics, health-informatics, anomaly/intrusion detection or information retrieval. Similarity functions that measure the resemblance between an ordered list of events has been used to classify sequences [Zhengzheng et al., 2010]. This technique has been applied to calculate similarities between alarm flood sequences [Kabir et al., 2013].

In the present work, it is assumed that similar alarm flood sequences are originated from the same kind of process abnormality. Pattern analysis is used to cluster similar alarm flood sequences in order to find the flood patterns that characterises the different abnormal situations encountered in a process plant.

In the proposed method, an algorithm that extracts the similarities between alarm flood sequences taking into account their time-stamp is used. This algorithm is the Modified Smith-Waterman algorithm (MSW) [Cheng et al., 2013] and it is shortly described in this chapter. Later, hierarchical clustering techniques used to obtain clusters of objects given the similarities between them are discussed.

2.1 Modified Smith-Waterman algorithm

The original Smith-Waterman algorithm was suggested by T.F. Smith and M.S. Waterman (1981). Its objective was to identify the longest homologous subsequences within a collection of large sequences. The algorithm finds "a pair of segments, one from each of two long sequences, such that there is no other pair of segments with greater similarity (homology)" [Smith and Waterman, 1981].

In alarms the time stamp should be taken into account. It is more important the proximity in time of the alarms than the exact order of occurrence.

In [Cheng et al., 2013] Y. Cheng et al. developed an algorithm based on Smith-Waterman to calculate a similarity index between two alarm flood sequences. They pointed out that it could occur that a group of linked alarms are triggered practically simultaneously and their order of appearance is random. Therefore, when analysing alarm sequences the exact order of occurrence of the alarms is not as important as their proximity in time.

The algorithm presented in the paper [Cheng et al., 2013] will be refered as the Modified Smith-Waterman (MSW) algorithm in this thesis. Its main advantages against other algorithms is that it takes into account the time stamp of the alarm occurrence and it blurs the order of the alarms in the sequence when they occur close in time. This clear advantage of the MSW is the main reason for its use in the present work.

The Algorithm The Smith-Waterman algorithm identifies the local sequence alignment of a pair of sequences with the highest similarity. Before explaining the algorithm let's explain what a local alignment is. Given two symbolic sequences, a segment of each of them can be extracted. The number of elements of the segments can be equalized by inserting gaps ("-") in one or both of them. For each symbol of one of the segments there is a symbol at the same position in the other segment. This is called a local alignment of the two sequences.

For example, from a couple of symbolic sequences:

$Sequence1 : 3,2,5,6,3,4,1,2$
$Sequence2 : 7,6,3,4,5,2,3,1,2,5$

a pair of segments can be extracted:

$A : 2,5,6,3,4,1,2$
$B : 7,6,3,4,5,2$

an alignment of these segments could be:

$A_1 : -,2,5,6,3,4,1,2$
$B_1 : 7,6,-,3,4,5,-,2$

or

$A_2 : 2,5,6,3,4,1,2$

$B_2 : -,7,6,3,4,5,2$

In both cases the two aligned segments have the same length (8 for the first case and 7 for the second). Then, each symbol in one of the aligned segments has a corresponding symbol in the aligned segment of the other sequence for the same position. These couple of symbols are called symbolic pairs of the alignment. For instance, (2,6) or (5,-) are symbolic pairs of the local alignment $A_1 - B_1$.

For a pair of sequences there exist multiple possible alignments. Therefore, to identify the local sequence alignment that is the most similar, a scoring system is required. For each symbolic pair (x,y) a similarity score is calculated. In the original Smith-Waterman algorithm the score of a symbolic pair is:

- a negative number δ when the symbolic pair has a gap
- a positive number α (usually 1) when the symbols are the same
- a negative number μ when the symbols are different

The similarity score of the local sequence alignment is the sum of the similarity scores of its symbolic pairs. For instance, for the alignment:

$A_2 : 2,5,6,3,4,1,2$
$B_2 : -,7,6,3,4,5,2$

if we choose $\delta = -0.4$, $\alpha = 1$ and $\mu = -0.6$, the similarity score of the alignment is:

$$\delta + \mu + \alpha + \alpha + \alpha + \mu + \alpha = -0.4 - 0.6 + 1 + 1 + 1 - 0.6 + 1 = 2.4$$

Then the alignment with the highest similarity score is the optimal alignment and its score is the similarity score between the two sequences.

However, an alarm sequence is different from a symbolic sequence in that an alarm is defined by an alarm type and a time stamp. If the alphabet of alarm types is:

$\Sigma = 1,2,...,K$

An alarm sequence can be defined as:
$A = a_1, a_2, ..., a_M$
$a_m = (e_m, t_m), m = 1,2,...,M$

where e_m is the alarm type, an integer between 1 and K, and t_m is the time stamp.

In [Cheng et al., 2013] a new similarity score for a pair of alarm occurrences with time stamp is defined. The new similarity score uses two new concepts "time distance vector" and "time weight vector".

For each alarm occurrence in an alarm sequence a time distance vector is defined as follows:

$$\mathbf{d}_m = [d_m^1, d_m^2, ..., d_m^K]^T \text{, for } k = 1, 2, ..., K$$

$$d_m^k = \begin{cases} \min\limits_{1 \leq i \leq M} \{|t_m - t_i| : e_i = k\}, \text{if the set is not empty} \\ \infty, \qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{2.1}$$

d_m^k is then the time interval between the mth alarm occurrence and the closest alarm occurrence of type k. If the alarm sequence has no alarm of the type k then the value is ∞. And $d_m^{e_m}$ will always be 0.

The time weight vector is defined as follows:

$$\mathbf{w}_m = [w_m^1, w_m^2, ..., w_m^K]^T = [f(d_m^1), f(d_m^2), ..., f(d_m^K)]^T$$

where $f() : \mathbb{R} \rightarrow \mathbb{R}$ is a time weighting function with respect to the time distance. The function has to satisfy:

1. Monotically decerasing on the positive axis
2. $f(0) = 1, f(\infty) = 0$

In [Cheng et al., 2013] the authors use a scaled Gaussian function as the time weighting function for the first sequence, the same function is used in the present work:

$$f(x) = \mathrm{e}^{-x^2/2\sigma^2}$$

for the second sequence, the following function is used:

$$f(x) = \begin{cases} 0, \ if\, x \neq 0 \\ 1, \ if\, x = 0 \end{cases} \tag{2.2}$$

However in our case study some alarms that occur exactly at the same time. If this last function was used for the second sequence together with the Gaussian for the first, some alarm matching will be counted twice. In order to avoid this, the weighting vector will be computed as follows:

$$w_m^k = \begin{cases} 0, \; if \, e_m \neq k \\ 1, \; if \, e_m = k \end{cases} \tag{2.3}$$

In [Cheng et al., 2013] a new similarity score between two alarm occurrences $((e_a,t_a),(e_b,t_b))$ was defined as follows:

$$s((e_a,t_a),(e_b,t_b)) = \max_{1 \leq k \leq K} \{w_a^k \times w_b^k\}(1-\mu)+\mu \tag{2.4}$$

2.2 Agglomerative Hierarchical Clustering

The goal of a clustering analysis is to group items into different sets or clusters, maximizing the similarities among items within the same cluster while minimizing the similarities among items within different clusters [Cai et al., 2014].

Clustering analysis is used in many applications, for instance: data compression, analysis of gene expression data, anomaly detection, structuring results of search engines [Ackermann et al., 2012], improve the structure of network systems [Seema and Coyle, 2003], organisms classification [Williams, 1975]...

Agglomerative Clustering is one of the most widely used and earliest approaches. It is a bottom-up clustering process where each input item forms its own cluster at the beginning [Ackermann et al., 2012]. Before starting the clustering algorithm, similarities (or distances) between all possible pairs of items have to be known. In each iteration of the clustering process the two clusters that are the most similar (or have the least distance between them) are merged. This process continues until either just one cluster remains or when the similarities between the clusters are lower than a defined minimum.

The results of the agglomerative hierarchical clustering are usually shown by a dendrogram, which lists all the items and at what level of similarity they where merged. An example of a dendrogram is shown in Figure 2.1.

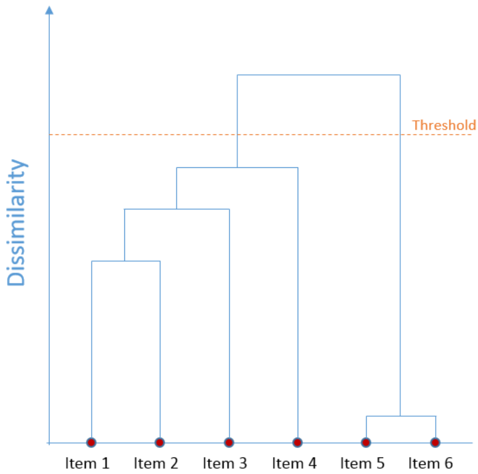


Figure 2.1 Dendogram

In the dendrogram shown, the vertical axis represents the dissimilarity between two clusters. Each fusion of two clusters is represented by the merging of two vertical lines (horizontal line), the vertical position at which the two vertical lines are merged into one gives the dissimilarity (or distance) between two clusters. The dashed horizontal line represents the threshold set for stopping the clustering algorithm. Looking at this dendrogram, one can see that $Item5$ and $Item6$ are the most similar items, and therefore the first ones merging. After this $Item1$ and $Item2$ merge together, then $Item3$ merges with the cluster containing $Item1$ and $Item2$. Later $Item4$ is added to this last cluster. Since the similarity between the cluster consisting of $Item5$ and $Item6$ has a similarity (dissimilarity) with the cluster formed by $Item1$, $Item2$, $Item3$ and $Item4$ lower (higher) than the threshold, these two clusters are not merged.

As mentioned before, in each iteration (when two clusters merge) the similarity between the new cluster and the rest is updated. There are different strategies to calculate the similarity between clusters. The most frequently used are the following:

- Single linkage
- Complete linkage
- Average linkage

In $single-linkage\ clustering$ the similarity of two clusters is the similarity of their most similar member. A and B being two clusters, the similarity between them is defined as:

$$S(A,B) = max\{S(a,b) : a \in A, b \in B\} \tag{2.5}$$

In $complete - linkage\ clustering$ the similarity of two clusters is the similarity between their most dissimilar members.

$$S(A,B) = min\{S(a,b) : a \in A, b \in B\} \tag{2.6}$$

In $average - linkage\ clustering$ the similarity of two clusters is the average of the similarities between their members.

$$S(A,B) = \frac{\sum\limits_{a \in A} \sum\limits_{b \in B} S(a,b)}{|A||B|} \tag{2.7}$$

where $|A|$ is the number of items in cluster A.

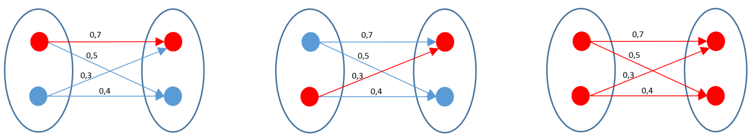


Figure 2.2 (a) Single Linkage, (b) Complete Linkage, (c) Average Linkage

For better understanding an example is shown in Figure 2.2. Two clusters with two items each are displayed. The numbers above the arrow represent the similarity index between the two items connected by the arrow. If $single - linkage\ clustering$ is used the similarity between the two clusters will be 0.7 (Figure 2.2a). If $complete - linkage\ clustering$ is used, the similarity between the clusters will be 0.3 (Figure 2.2b). Finally if $average - linkage\ clustering$ is used, the similarity will be calculated as follows:

$$S(A,B) = \frac{0.7+0.5+0.3+0.4}{4} = 0.475$$

A simple HAC (Hierarchical Agglomerative Clustering) algorithm is shown below:

Input: $N \times N$ similarity matrix S, where N is the number of items and a threshold τ.

1. Assign a cluster for each item
2. Find the most similar clusters and merge them

3. Compute the similarity index between the new cluster and the rest and update the similarity matrix. The size of S becomes $N-1 \times N-1$

4. Repeat step 2 to step 3 until $maxS > \tau$ or $N = 1$

3

Root-cause analysis based on Process Data

As stated in Chapter 1, nowadays industrial plants present a high level of interconnectivity due to the multiple material, information and energy connections. Presence of a plant-wide disturbances is a common situation, however their cause is difficult to diagnose [Yuan and Qin, 2014]. During the past few years several methods for process data analysis have been developed in order to find the root-causes of plant-wide disturbances. In [Thornhill, 2005] a method that analyses process signals to distinguish the root-cause of an plant-wide oscillation from the propagated secondary oscillations is suggested. T. Yuan et. al propose Granger causality and spectral Granger causality to provide cause and effect relationships between time series. Margaret Bauer et al. present in [Bauer et al., 2007] a method used to identify the direction of propagation of disturbances based on the probability density function measuring the directionality of variation.

In this thesis a Plant-wide Disturbance Analysis software tool (PDA) developed at ABB is used to perform a data-driven root-cause analysis. A brief description of the tool and the methods on which it is based is presented in this chapter.

However one must take into account that even though the methods used by PDA are regarded as techniques that analyse causality, they may not capture true causality since they are pairwise methods [Fang Yang, 2012]. In this thesis the expression causality analysis is used since it is the term used in both industrial and research communities, however one should keep in mind that it is actually a precedence analysis.

3.1 PDA

PDA is a software tool developed by ABB used for plant-wide disturbance analysis. It is based on the methods representing the state of the art for the analysis of upsets and disturbances in process plants.

The data-driven analysis performed by PDA has basically four steps:

1. Data preprocessing
2. Data filtering
3. Clustering
4. Root cause analysis

In the first step, data that is not suitable for the analysis is removed, these can be compressed data or data with missing values or non numerical entries. In the second step, the user can apply filters available in the tool to select the desired signal spectral range for the analysis. The different signal tags included in the analysis are clustered in step three. Finally, root cause analysis is performed for the different clusters in step four.

PDA uses two different methods to cluster signals based on:

- Similarity of oscillation indices
- Principal component analysis of spectral data

Regarding root-cause analysis the tool presents two sections of results:

- Non-linearity index
- Time delay of cross correlation function and transfer entropy

The different clustering and root-cause analysis methods are briefly described in the following subsections.

3.1.1 Oscillation detection

PDA is based on the method presented in [Thornhill et al., 2003a] to detect oscillations in signals and cluster them. The method is based on the zero crossings of the autocovariance function (ACF). The main advantage of using ACF instead of using directly the time trend relies on the fact that (white) noise that can cause spurious zero crossings in the latter is removed when using the ACF. This is due to the fact that the ACF of the white noise is theoretically zero for lags greater than zero.

Those signals whose ACF have regular oscillations and whose average oscillation periods are similar enough are cluster together. For additional information about the method and the clustering algorithm see [Thornhill et al., 2003a].

3.1.2 Spectral analysis

In PDA, principal component analysis is applied to the power spectra of the signals in order to identify their features. Power spectra is invariant to time delays caused by plant dynamics, therefore it is possible to cluster signals having similar spectral components even if they are delayed in time. Defining $\mathbf{X}$ as a matrix whose rows are the power spectra of the different measurements for a range of frequencies up to the Nyquist frequency (one half of the sampling period). PCA decomposition reconstructs the matrix $\mathbf{X}$ as a sum over a basis functions w_i which are spectrum-like functions.

$$\mathbf{X} = \sum \mathbf{t}_i \cdot \mathbf{w}_i + \mathbf{E} \tag{3.1}$$

where $\mathbf{E}$ is the error matrix that captures the variation of $\mathbf{X}$ that is not covered by the mode and $\mathbf{t}_i$ are the scores.

PCA maps each row vector of $\mathbf{X}$ to a specific coordinate in the score plot. Then those signals with common spectra signatures will be isolated since they will be close in the score plot.

3.1.3 Non-linear root-cause

The goal of the non-linear root-cause method is to isolate the signal closest to the root-cause within a cluster of oscillating signals.

The idea behind this technique is that oscillations are commonly caused by the presence of non-linearities in the system (limit cycles). These non-linearities are attenuated as they propagate through the plant since the assets in the plant act as a low pass filter. Therefore, the measurement closest to the root-cause will have a higher non-linearity index.

PDA computes a numerical non-linearity index for each signal [Thornhill et al., 2003b]. Then the signals within the cluster are presented in an ordered list with the measurement with the highest non-linearity index (closest to the root-cause) on top.

3.1.4 Causality analysis

PDA allows the use of two different techniques to perform causality analysis, these are, *Time delay analysis* and *Transfer entropy*.

Time delay analysis

This method detects the precedence relationship between two signals using the cross-correlation function (CCF).

The idea here is that when a disturbance spreads in a process plant, several signals are affected. However not all signals are affected simultaneously as the disturbance needs some time to propagate form one signal to another. Therefore, the earlier a signal is affected by the disturbance the closer it is to the root-cause.

The CCF measures similarities between two time trends for different time lags between them. The time lag for which the cross-correlation reaches its maximum is the time lag for which the two time trend are best aligned. Then if two signals are delayed, this time lag will correspond to the time delay between them.

Transfer Entropy

Transfer entropy is a data-driven method proposed by Schreiber [Schreiber, 2000] which uses conditional probability density functions to identify precedence relationships between process variables.

These relationships are used to determine the direction in which a disturbance propagates between the different signals and to help finding the root-cause of this disturbance.

Other methods analyse the time delay between two measurements to identify the directionality. However, time lags are difficult to identify when the signals are oscillatory or noisy. Furthermore, if the delay is smaller than the sampling time of the process data, the causality relationship cannot be determined. Transfer entropy gives solution to these problems, it uses transition probabilities to estimate the amount of information transferred from one signal to another.

Some steps are followed to extract the causal relationships with this method:

1. Estimation of the joint PDFs (Probability Density Function), using a Kernel estimator
2. Calculate transition probabilities
3. Calculate transfer entropy values
4. Construction of the causal map

Estimation of joint PDFs

For the computation of the transfer entropy a probability density function from the time series corresponding to the process signals must be estimated. In this case a Gaussian Kernel function is used:

$$\hat{p}(x,y) = \frac{1}{N}\sum K(x-x_i, y-y_i)$$

Where K is the Gaussian Kernel function, N is the number of samples in the time series x and y, and $\hat{p}(x,y)$ is its joint probability estimation.

Transition Probabilities

The transition probability can be obtained through joint PDFs as it follows:

$$p(x_{i+h}|\mathbf{x}_i, \mathbf{y}_i) = \frac{p(x_{i+h}, \mathbf{x}_i, \mathbf{y}_i)}{p(\mathbf{x}_i, \mathbf{y}_i)}$$

$p(x_{i+h}|\mathbf{x}_i, \mathbf{y}_i)$ denotes the probability that a future value x_{i+h} has a specific value when past values $\mathbf{x}_i = [x_i, x_{i-\tau}, ..., x_{i-(k-1)\tau}]$ and $\mathbf{y}_i = [y_i, y_{i-\tau}, ..., y_{i-(k-1)\tau}]$ are known.

Transfer entropy

The measure of transfer entropy is defined by Schreiber [ref 5] as follows:

$$t(x|y) = \sum p(x_{i+h}, \mathbf{y}_i, \mathbf{x}_i) \cdot log\frac{p(x_{i+h}|\mathbf{y}_i, \mathbf{x}_i)}{p(x_{i+h}|\mathbf{x}_i)}$$

The logarithm represents a comparison between the probability of x_{i+h} knowing the past observations **x** and **y** (numerator) and the probability of x_{i+h} knowing the past values **x** (denominator). If knowing past values of y does not affect the probability, that means that no information is transferred from y to x. In this case numerator and denominator will be the same and the logarithm will become, and then $t(x|y)$ will become 0.

The causality is obtained when comparing the information transferred from x to y with the information transferred from y to x.

$$t_{x\to y} = t(y|x) - t(x|y)$$

If $t_{x\to y}$ is positives that means that more information is transferred from x to y, and therefore x caused y. If $t_{x\to y}$ is negative, then information is transferred from y to x, and therefore y caused x. If $t_{x\to y}$ has a value close to 0, then the causality cannot be deduced.

Transfer entropy is a quantitative process history-based method that allows us to identify the direction of propagation of disturbances. Its main advantages is that it is a model free method and that it can determine directionality even in absence of time delay.

4

Connectivity analysis based on Plant Topology

On the last three decades fault diagnosis in the chemical industry has been the focus of many research groups [Gao et al., 2010]. Qualitative model based methods, i. e., methods based on some fundamental understanding of the process that captures the structure of a system, have been used to identify the source of a disturbance.

Among these methods, graphs have drawn more attention and have been improved in the past years. In the present work, graphs are used to represent the plant connectivity and to automatically explore relationships between assets to validate the root-cause analysis results obtained from a pure data-driven analysis.

This chapter describes the use of graphs for exploring the connectivity of a process plant. First, a short introduction to graph theory is provided. Later, the application of graph theory to causality analysis verification is explored. Finally, the chapter looks into the concept of intelligent P&IDs (Process and Instrumentation Diagrams) and the problem of transforming legacy topology data into computer interpretable formalized topology models.

4.1 Graphs in fault diagnosis

Physical systems can be represented qualitatively with graphs. Directed graphs (digraphs or DG) and signed directed graphs (SDG) are used to capture connectivity relationships between elements and have been widely applied to chemical process fault diagnosis [Venkatasubramanian et al., 2003]. Directed graphs are represented as graph nodes in which the precedence relationship between nodes is represented by directed arcs. In SDGs the arcs between nodes not only define the direction of the precedence relationship but also the sign of this relationship. The arcs have a positive or negative sign attached to them depending on whether a change on one of the nodes results in a change in the same direction or opposite direction respectively in the other node.

4.1.1 Graph Fundamentals

A graph $G = (V, E)$ consists of a set of objects $V = v_1, v_2...$ called vertices or nodes and a set $E = e_1, e_2...$ whose elements are called edges. Each edge is identified with an unordered (undirected graphs) or ordered (directed graph) pair of vertices (v_i, v_j). In the case of signed directed graphs, a sign "+" or "-" representing positive or negative influence respectively is assigned to the edge.

Figure 4.1 shows examples of an undirected graph (a) and a directed graph (b).

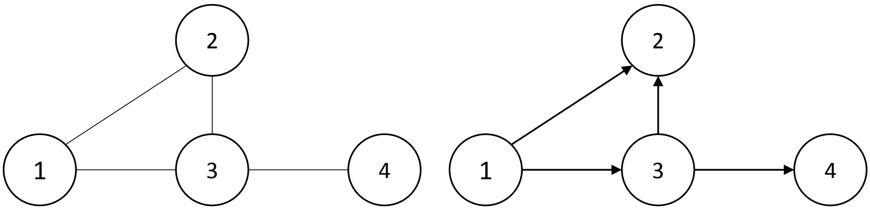


Figure 4.1 Graphical representation of an undirected graph (a) and a directed graph (b).

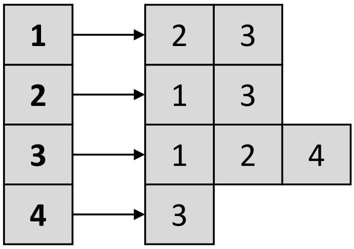
There exist two standard ways of representing undirected and directed graph $G = (V, E)$ in a computer: using adjacency lists or a connectivity matrix.

The adjacency list representation of a graph consists of an array with $|V|$ lists, one for each vertex or node in the graph. The list corresponding to a vertex u contains all vertices v for which exists an edge $(u, v) \in E$. Figure 4.2 (a) is an adjacency list representation of the undirected graph in Figure 4.1 (a) and Figure 4.3 (a) is an adjacency list representation of the directed graph in Figure 4.1 (b).

The connectivity matrix, assuming $V = 1, 2, ..., |V|$, consists of a $|V| \times |V|$ matrix $A = (a_{ij})$ such that

$$a_{ij} = \begin{cases} 1, \; if\,(i,j) \in E \\ 0, \; otherwise \end{cases} \tag{4.1}$$

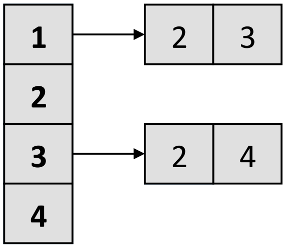
Figures 4.2 (b) and 4.3 (b) are the connectivity matrices of the undirected and directed graphs in Figures 4.1 (a) and (b) respectively.



$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

(a) Undirected Graph (b) Directed Graph

Figure 4.2 Two representations of the undirected graph in Figure 4.1. (a)Adjacency list (b) Connectivity matrix



$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(a) Undirected Graph (b) Directed Graph

Figure 4.3 Two representations of the directed graph in Figure 4.1. (a)Adjacency list (b) Connectivity matrix

4.1.2 Graphs and causality analysis in process plants

For the purpose of diagnostic reasoning, to validate the root-cause of a disturbance propagating in a process plant, a directed graph representing the connectivity of the plant can be built with qualitative knowledge of the plant. Each node represents an asset in the plant (indicators, controllers, vessels, valves...) and the edges symbolize the existence of connection between the assets. An edge $e_k = (v_i, v_j)$ symbolizes that there is a material or information connection from asset *i* to asset *j*.

For example, consider a quenching process as shown in Figure 4.4, where *QC* is a quenching column, *V1* a valve regulating the inflow of quenching liquid into the quenching column, *T1* the tank where the quenching liquid is kept. The hot gas enters the quenching column where it is cooled by direct contact with the quenching liquid. *TC* is the controller measuring the temperature at the output of the quenching column and opening/closing *V1* accordingly. Solid lines represent pipes, while dashed lines represent information links.

A DG is built from process knowledge, Figure 4.5 (a). Pipes are not considered nodes for simplicity. They will be seen as material links between assets. In case a controller or an indicator is attached to a pipe, a link between the measurement point and the asset upstream the pipe is set instead. Furthermore, material links are considered bidirectional, while information links are considered unidirectional. Even though one asset is located downstream of another, a change of certain properties such as pressure or temperature in the asset downstream can affect the asset upstream. As for the information links, an indicator is affected by the asset whose property is measuring. However, the a change in the state of the indicator (such as malfunctioning) does not affect the asset. Therefore, information connections are considered unidirectional.

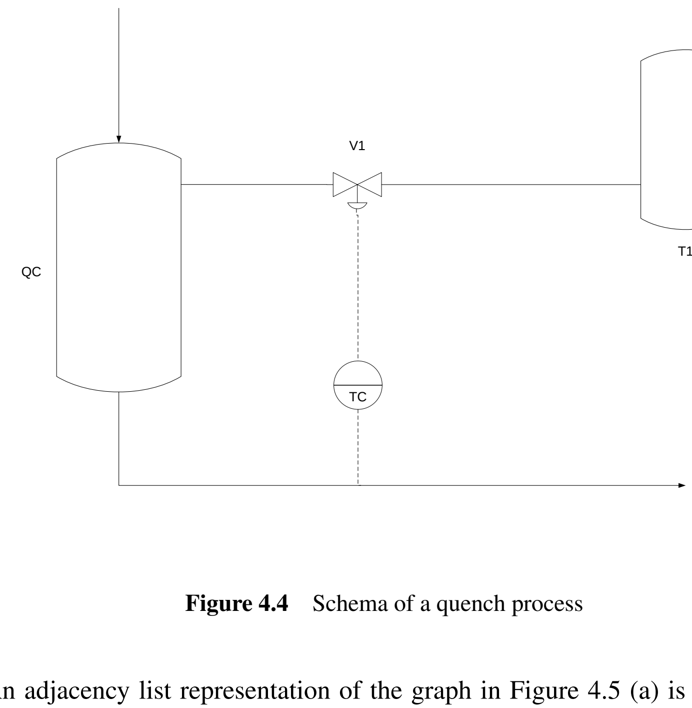
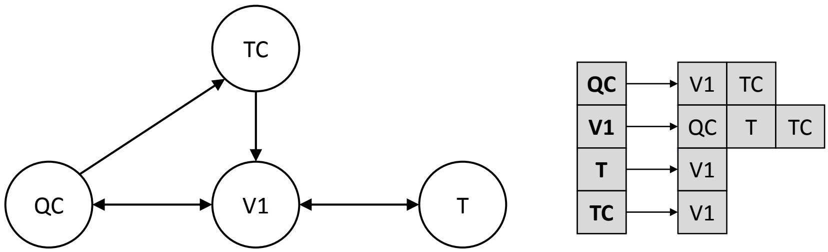


Figure 4.4 Schema of a quench process

An adjacency list representation of the graph in Figure 4.5 (a) is presented in Figure 4.5 (b). In this work, adjacency lists have been chosen over connectivity matrices since regularly plant process graphs are sparse, i.e., $|E|$ is considerably less than $|V|^2$. For sparse graphs, adjacency lists require less storage space because they onliy need to represent those connections that not present.



(a) (b)

Figure 4.5 (a) Graphical representation of schema in Figure 4.4. (b) Adjacency list representation of the graph in (a)

Once a graph that captures the plant connectivity is built, the next step is to explore the information contained in it. Two well established graph-search algorithms are *Breadth-first search* and *Depth-first search*.

Breadth-first search This algorithm explores exhaustively a graph by using a first in first out (FIFO) approach in a queue storing nodes. The algorithm starts at a specified starting node *s*, it places it into the queue and then it explores all its adjacent nodes and adds them to the queue, then the starting node is removed from the queue. Later all unexplored nodes adjacent to the next node in the queue are stored in the queue and the node is removed. This procedure is repeated untill the queue is empty.

Depth First Search Depth first search uses instead a last in first out (LIFO) approach. The algorithm places in the queue the starting node, then one of the undiscovered adjacent nodes is explored exhaustively until a node that has no unexplored adjacent nodes is reached. Finally, the search backtracks to the previous node visited to explore any of its undiscovered adjacent nodes.

Following [Thambirajah et al., 2009], plant connectivity is used to validate the result from a pure data-driven analysis. The suggested root-causes obtained from the data analysis are specified as starting nodes and all possible paths starting from the hypothetical root-causes are explored. In this way, it can be determined whether there is a feasible path between the suggested root-cause and the other disturbed elements. If just a small proportion of the disturbed assets is found to be connected to one of the root-cause suggestions, chances are high that this element is not the root-cause. Then, spurious results from the data-driven analysis are ruled out [Thambirajah et al., 2009].

4.2 Intelligent P&IDs

P&ID stands for Piping and Instrumentation Diagram. P&IDs play a crucial role in the design and maintenance of process plants. It is a drawing of the process showing the instrument details and their connectivity as well as the control schemes. A vast amount of engineering data and information is stored in these graphical representations. However, an automatic extraction of this information is hard to manage directly from the graphical representation.

In order to facilitate data exchange across different systems new formats and schemas have been developed. XML (eXtensible Markup Language) is a markup language for documents containing structured information created for this purpose. XML provides a flexible way to create common information formats that can be used by any group of individuals that want to share information in a consistent and platform independent manner. XML was rapidly adopted as the meta data format for electronic commerce applications. It provided a "common ground" for independent entities to easily exchange data to conduct business [Li, 2000]. Examples of it are the construction industry for data interchange related to resources, activities and transportation (AecXML,TransXML) [AGDAS and ELLIS, 2010] and the chemical industry for buying, selling and deliverying chemicals (Chem eStandard).

XML has been recently used in the so called intelligent P&IDs (iP&IDs). iP&IDs are representations of the plant in a computer available format that include not only the traditional plant drawings but also additional information about the assets, their connectivity and their layout. CAEX (Computer Aided Engineering eXchange) is a vendor independent object-oriented data exchange format based on the XML schema that has been used in the past years for structuring plant topology data in a flexible and expandable way (see e.g. [Schleburg et al., 2013], [Thambirajah et al., 2009], [Schleipen et al., 2008]). It has been developed in cooperation with the Chair of Process Control Engineering of the RWTH Aachen and the Research Center of ABB, and it is standardized in IEC 62424 [IEC, 2008].

In order to efficiently define objects and their connectivity, CAEX adopts three libraries: *SystemUnitClassLib*, *RoleClassLib* and *InterfaceClassLib*. Plant elements are represented as nodes called *InternalElements*. Each *InternalElement* is an instance of a *SystemUnitClass*, which defines a template of a specific plant component, and has a reference to a role class, which defines functional attributes and interface information. The material, energy or information connections between plant elements are defined as *Interfaces* and the different types of interfaces are contained in the *InterfaceClassLib*. Finally, the plant is described in the *InstanceHierarchy*, where the different classes are instantiated, referenced and interconnected to shape the topology model of the plant.

In the following simplified example, a section of an *InstanceHierarchy* describes a system consisting of a temperature indicator connected to a quenching column:

```
<InternalElement Name="QuenchingColumn"
                 RefBaseSystemUnitPath="SystemUnitLib/Vessel"
                 ID="QuenchingColumn">
  <Attribute Name="_Height">
    <Value>400</Value>
  </Attribute>
  <Attribute Name="_Diameter">
    <Value>312</Value>
  </Attribute>
      ...
  <ExternalInterface Name="Info_MeasurementPoint1"
                     RefBaseClassPath="InterfaceLib/InformationConnector"
                     ID="Info_MeasurementPoint1" />
  <RoleRequirements RefBaseRoleClassPath="RoleLib/Volume" />
</InternalElement>
```

```
<InternalElement Name="TI1"
                 RefBaseSystemUnitPath="SystemUnitLib/ContinuousSensor"
                 ID="TI1">
  <Attribute Name="_MeasuringUnit" />
   <Value>"K"</Value>
  </Attribute>
  ...
  <ExternalInterface  Name="Info_LogicalNozzle1"
                      RefBaseClassPath="InterfaceLib/InformationConnector"
                      ID="Info_LogicalNozzle1" />
  <RoleRequirements RefBaseRoleClassPath="RoleLib/Sensor" />
</InternalElement>
```

```
<InternalLink Name="Link1"
              RefPartnerSideA="Info_MeasurementPoint1"
              RefPartnerSideB="Info_LogicalNozzle1"
              ID="Link1" />
```

The first element of the example is an *InternalElement* representing the quenching column. The internal element is defined by *Name*, *ID* and its *SystemUnitLib* class. Later, attributes of the quenching columns (height, diameter...) and its role class are specified. The *InternalElement* QuenchingColumn is an instance of the class Vessel and has a reference to the role Volume.

The next *InternalElement* is the temperature indicator. In this case, the element is an instance of the class ContinuousSensor and has a reference to the role Sensor.

The information connection between the two elements is performed by defining an *ExternalInterface* of the type InformationConnector inside the *InternalElement* describing the assets and then connect them with an *InternalLink*. The *InternalLink* describes the directional link between the assets. *RefPartnerSideA* refers to the element from which the link is originated and *RefPartnerSideB* refers to the element where the link terminates. In this case the link goes from Info_LogicalNozzle1 (QuenchingColumn) to Info_MeasurementPoint1 (TI1), representing an information connection going from the quenching column to the sensor.

4.2.1 Conversion of P&IDs into iP&IDs

The research community sometimes assumes the availability of plant topology descriptions in a computer interpretable formalized representation. However, the reality is far from this assumption. Usually P&IDs are still exchanged on a paper base or portable document format (.pdf) where no additional information besides the graphical symbols and arcs is included. The reason for this could be that the process plant was designed using traditional CAD drawing tools and then there is no additional information available, or that the companies fear to loose the know-how to other companies.

Finding methods to automatically or semi-automatically upgrade legacy data to the new formats has become a relevant matter in the process industry. That is the reason why several CAD vendors have put effort on developing software tools in order to facilitate the conversion. SmartPlant® offers a migration utility called Import Assistant that supports the conversion of P&IDs created in AutoCAD, MicroStation and PDS®2D to SmartPlant P&IDs. CADWorx also offers a tool that makes easier to connect legacy P&ID's components to project databases.

In this thesis the availabilitz of an intelligent P&ID in CAEX format is assumed. In case an intelligent P&ID is not provided a transformation tool will be used to create the computer readable topologz model.

5

The Method

This chapter describes an original approach to reduce alarm flood periods in process plants. The steps of the method will be detailed in the sections of this chapter.

The proposed approach relies on two assumptions:

- Alarm floods are the result of an abnormality propagating in the process through material, energy or control connections.
- If two alarm flood sequences are similar enough, then they originate from the same source.

Regarding the first hypothesis, it was stated before in *Chapter 2* that the focus of this thesis is on the reduction of consequential alarms. The reduction of repetitive alarms or standing alarms have been widely investigated, and methods to remove them are already available and well established (e.g. [EEMUA-191, 2007],[Hugo, 2009],[Ahnlund et al., 2003]). In this thesis the alarm floods of interest are those caused by an irregularity spreading through the plant due to its high level of interconnectivity.

As for the second hypothesis, if two alarm flood sequences share a common pattern and a big portion of the alarm occurrences in the alarm floods are in this pattern, it is very likely that both alarm floods are the result of the same process abnormality.

The objective of the proposed approach is to reduce alarm floods in process plants by grouping consequential alarms originating from the same cause and to give a suggestion of the "causal" alarm.

The method consists of five basic steps:

1. Remove chattering alarms
2. Identify alarm flood time-periods
3. Cluster alarm flood sequences
4. Mapping
5. Root-cause analysis

The first three steps correspond to the alarm log analysis stage. In the first step chattering alarms will be removed from the alarm log. In the second step, alarm flood time periods are isolated and alarm flood sequences are constructed. In the third step, similarity indices between sequences are computed and the alarm sequences are clustered according to them. Each cluster represents a process abnormality and the alarm sequences in the cluster are the periods where the abnormality occurred. The forth step of the method sets the connection between the alarm, the process signals and assets' tag names. Finally, on the fifth step a precedence analysis is performed to isolate the "causal" alarm in the alarm flood sequence.

5.1 Remove Chattering alarms

In industrial process alarm management systems of industrial processes there usually exist a large amount of nuisance alarms. The most common type of "nuisance alarms" are the so called "chattering and repetitive" alarms [EEMUA-191, 2007]. Chattering alarms are defined by the industrial standard ISA-18.2 (2009) as alarms that "repeatedly transition between the alarm state and the normal state in a short period of time". About 10 to 60% of the alarm occurrences are due to chattering and repetitive alarms [Rothenberg, 2009]. The most common causes of chattering alarms are [Kabir et al., 2013]:

- The presence of noise in the process measurements.
- The process variable corresponding to the alarm is operating at a critical value very close to the alarm limit.

If chattering alarms are not removed prior to the alarm flood analysis, we may find time intervals with a large amount of alarms where most of the alarm occurrences are chattering alarms, as exemplified in Figure 5.1.

The X axis of the plots in Figure 5.1 represents time, and is divided into intervals of 10 min. The Y axis shows the number of alarm occurrences within each interval of 10 min. In Figure 5.1, left side the removal of chattering alarms is not performed. While for the plot on the rightside, chattering alarms are removed. It can clearly be seen that on this example, chattering alarms have a great contribution on the count of alarms, and these nuisance alarms are causing some of the intervals to have a relatively high number of alarms.

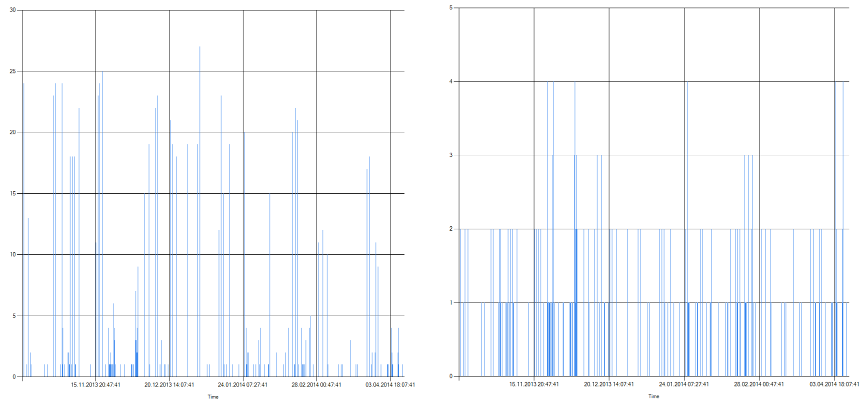


Figure 5.1 Chattering alarm example (a) Original alarm log and (b) Alarm log after chattering alarm removal

In the ISA 18.2 it is not specified whether chattering alarms should be included within the 10 alarms per 10 minutes threshold that defines an alarm flood. However, since the present work focuses on analysing consequential alarms, chattering alarms are removed before proceeding with the analysis. The alarm analysis focuses on alarm floods involving consequential alarm sequences.

Methods for chattering alarms removal have been widely investigated by both academic and industrial communities. Several methods for handling this type of nuisance alarms have been developed. In [EEMUA-191, 2007], it is suggested to use filtering, dead-band, delay timer and shelving as mechanisms to remove chattering alarms. More complex methods have been proposed, Hugo [Hugo, 2009] uses time-series modelling to estimate alarm deadbands to remove chattering alarms. In [Naghoosi et al., 2011] the authors propose a method to estimate the optimal alarm threshold taking into account the dead-band and the process variable characteristics in order to reduce the amount of chattering. In [Wang and Chen, 2014], the authors present two rules to detect chattering alarms caused by random noise and suggest a method for online removal using m-sample delay timer.

Developing a method for reduction of chattering alarms is outside the scope of this thesis. However, it is a required step to isolate the time intervals of interest. A simple method will be used for this purpose. The method used is the one suggested in [Ahmed, 2011].

In order to remove chattering alarms, a period of time will be defined and if the time between two alarm occurrences of the same type is lower than the chosen interval, the second alarm occurrence will be removed. The interval length chosen for this thesis is 10 min, based on the definition of alarm flood.

The algorithm implemented in our tool for removing chattering alarms is described in Figure 5.2.

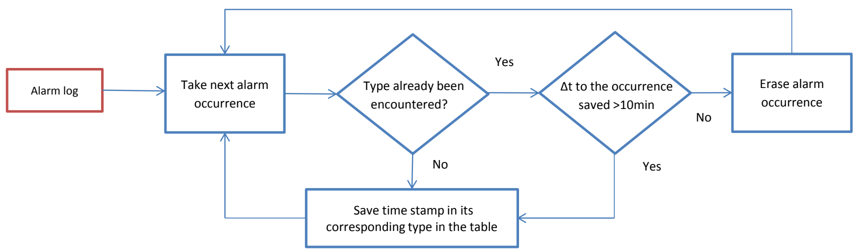


Figure 5.2 Chattering alarms' removal algorithm

A table with two columns is created. In the first column all alarm types are listed and in the second the time stamps of the last alarm occurrence of the corresponding type that has not been removed are saved. Each alarm occurrence in the alarm log is analysed. First, if its alarm type has not be encountered yet, its time stamp is saved in the second column in the row corresponding to its alarm type. If the alarm type of the alarm occurrence that is being analysed has already been encountered and if the difference in time between its time stamp and the time stamp saved in the table is smaller than 10 min, the alarm occurrence is removed. Otherwise the time stamp of that alarm type in the table is substituted by the alarm stamp of the current occurrence, and the alarm occurrence is not erased from the alarm log.

5.2 Identification of alarm flood periods

Once chattering alarm occurrences are filtered out, the next step is to isolate those time-periods where alarm floods occur and construct the corresponding alarm flood sequences.

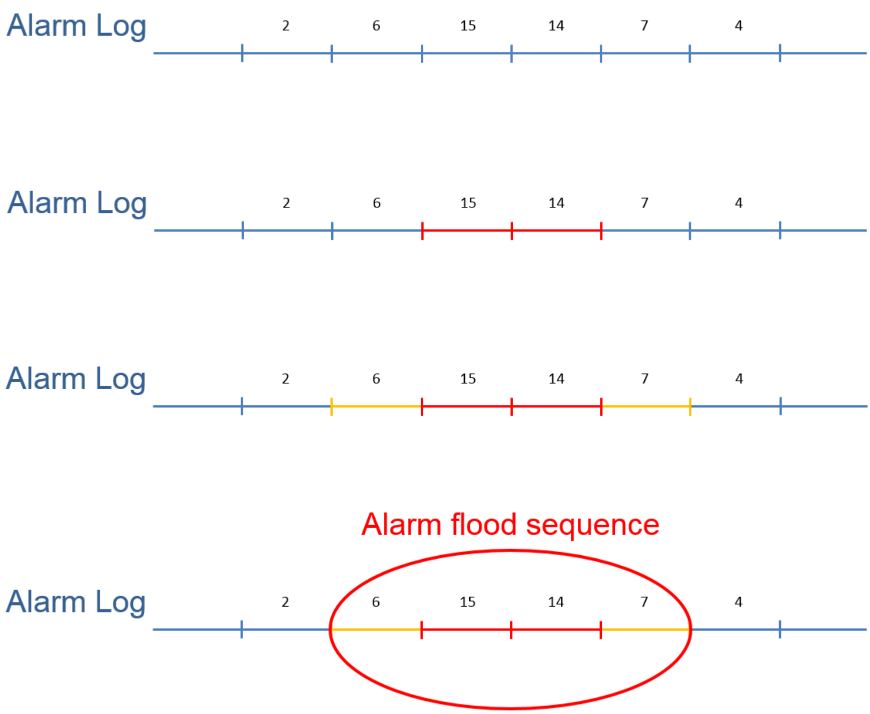
The alarm log is divided into intervals of 10 min. And those intervals with more alarm occurrences than a specified threshold are highlighted. The threshold (τ) will be chosen taking into account that an alarm flood is defined as more than 10 alarms per 10 min and operator. In process plants one can find alarm floods that take few minutes and others that last several hours. In order to identify an alarm floods regardless of its duration consecutive intervals with more alarm occurrences than the specified threshold are merged.

Since the alarm log is discretised, it can occur that the beginning or the end of an alarm flood sequence is cut out. For instance, considering a threshold equal to 10, an alarm flood sequence consisting of 20 alarms triggering within 10 min could be split apart when the log is divided into intervals of 10 min. The first 5 alarms might be contained in one interval and the other 15 in the following. If just those intervals with more than 10 alarms are considered, the first 5 alarms will not be taken into account, then part of the alarm sequence will be lost. For this study it is very important not to lose part of the sequence neither at the beginning, where the alarm connected to the root-cause of the alarm flood might be, nor at the end, where extra consequential alarms that can influence the alarm sequence clustering in step three (see Chapter 3). For this reason, the interval before and after the consecutive intervals with more than τ alarms are also taken into consideration.

The alarm flood sequences are built with the alarm occurrences within:

- Consecutive intervals with more than τ alarms occurrences.
- The time interval before.
- The time interval after.

An example of this process is shown in Figure 5.2 (τ = 10). In the upper part of Figure 5.2 a section of an alarm log is displayed. The numbers above the intervals represent the number of alarm occurrences in the intervals. In this section two consecutive intervals have more than 10 alarms, one containing 15 alarms and the other 14. Those two intervals are considered to build the alarm flood sequence. Then the intervals before and after are added (depicted in orange in the picture). The alarm occurrences within these four intervals form the alarm sequence corresponding to this alarm flood episode.



5.3 Cluster alarm flood sequences

Once the alarm flood periods are isolated and the alarm flood sequences are constructed, the next step is to cluster similar alarm floods sequences.

A similarity index is first computed for each pair of sequences. The similarity index used in the present work is the one described in [Cheng et al., 2013]. As stated above, when matching patterns in alarm sequences the proximity in time of the alarms is more important than the exact order of occurrence because of uncertainties. The MSW algorithm, summarized in Chapter 3, takes into account these uncertainties by blurring the order of those alarms that occur close in time.

The main issue when using the Modified Smith Waterman algorithm is the computation time. The time required to calculate the similarity index between two sequences can be greater than 10 seconds, depending on the sequence length. Computing the similarity index of all possible combinations of alarm flood sequences within an alarm log of half a year can easily take more than a day.

In order to overcome this problem, an initial stage is added where another algorithm that is less precise but less time consuming can be used for pre-filtering. A similar idea is used in [Ahmed, 2011]. The aim is to filter out those combinations of alarm sequences that have low similarity, so that the MSW index is just calculated for those combinations that are similar "enough".

A similarity matrix is built using these MSW similarity indexes. Later an agglomerative hierarchical clustering algorithm is used to group similar alarm sequences. The work flow of the clustering step is shown in Figure 5.3 below and is detailed in the sequel of this section.

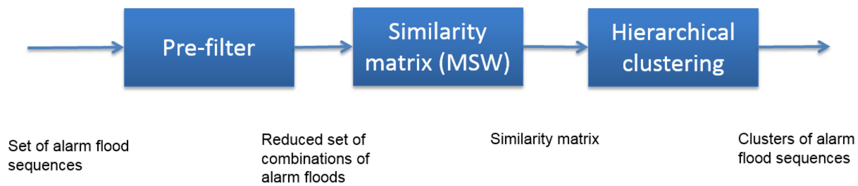


Figure 5.3 Clustering of alarm flood sequences workflow

5.3.1 Pre-Filter

Given a pair of alarm sequences, these sequences are considered similar if they have a high ratio of alarm types in common. If two alarm sequences have few alarms in common the computation of the MSW similarity index is discarded. These combination pairs can be directly assigned a low similarity index.

In [Kabir et al., 2013], the Jaccard distance is used for this purpose. Let A and B be two alarm sequences, the Jaccard distance between A and B is calculated as it follows:

$$J(A,B) = \frac{a+b}{a+b+c} \tag{5.1}$$

a being the number of alarm types in A that B does not contain. b the number of alarm types in B that A does not contain and c the number of alarms types contained in both sequences.

A threshold is set for the pre-filter. If the calculated Jaccard distance is greater than the threshold value, this combination will be pre-assigned a high distance value.

For the method described in this work, the pre-filter index is modified due to two reasons. First, as the MSW algorithm calculates a similarity index not a distance index, it is more convenient to use a similarity index for the pre-filter too. Second, as mentioned above, to decide whether a pair of alarm sequences is filtered out or not, a threshold must be set. The pre-filter similarity index chosen makes easier the threshold selection as explained below.

In the *Clustering alarm flood sequences* step of this method, two thresholds must be set: the pre-filter threshold and the clustering threshold. The first threshold one decides whether a pair of alarm sequences is filtered out or not, the second sets the stopping criteria of the clustering algorithm (i.e. if the maximum similarity between clusters at a certain iteration in the clustering method is lower than the threshold, the algorithm stops). The choice of these threshold values is interrelated: for a specific clustering threshold if the pre-filter threshold is set too high there may be combinations of alarm sequences that have a MSW similarity index big enough for being clustered, but their pre-filter similarity index is too low to pass the pre-filter. To avoid this situation, one possibility could be to set the pre-filter threshold very low. However, in this case, the computational time increases and the benefits of using a pre-filter are lost. An alternative solution is to make the pre-filter and the MSW similarity indexes comparable. If one could insure that $SI_{prefilter} \geq SI_{MSW}$, then for a given clustering threshold if the pre-filter threshold is set to be equal or smaller than the clustering threshold, all pairs of alarms that have a MSW similarity index high enough to be clustered together, will pass the pre-filter.

To illustrate this, assume the clustering threshold is set to a value of 0.5 and two alarm sequences have a MSW similarity index of 0.6 (and should therefore should be clustered). Setting the pre-filter threshold to a value of 0.5 or lower assures that this pair is not filtered out. Let be α the clustering threshold and β the pre-filter threshold, the following relation holds:

$$\beta \leq \alpha = 0.5 \leq SI_{MSW} = 0.6 \leq SI_{prefilter}$$

Given two alarm sequences A and B a similarity index that meets this requirement is:

$$SI_{prefilter}(A,B) = \frac{|A \cap B|}{max(|A|,|B|)} \tag{5.2}$$

The numerator represents number of common alarm occurrences in both sequences. And the denominator is the number of alarm occurrences of the longest sequence (the similarity index is normalized between 0 and 1).

The pre-filtering substep is summarized in Figure 5.4.

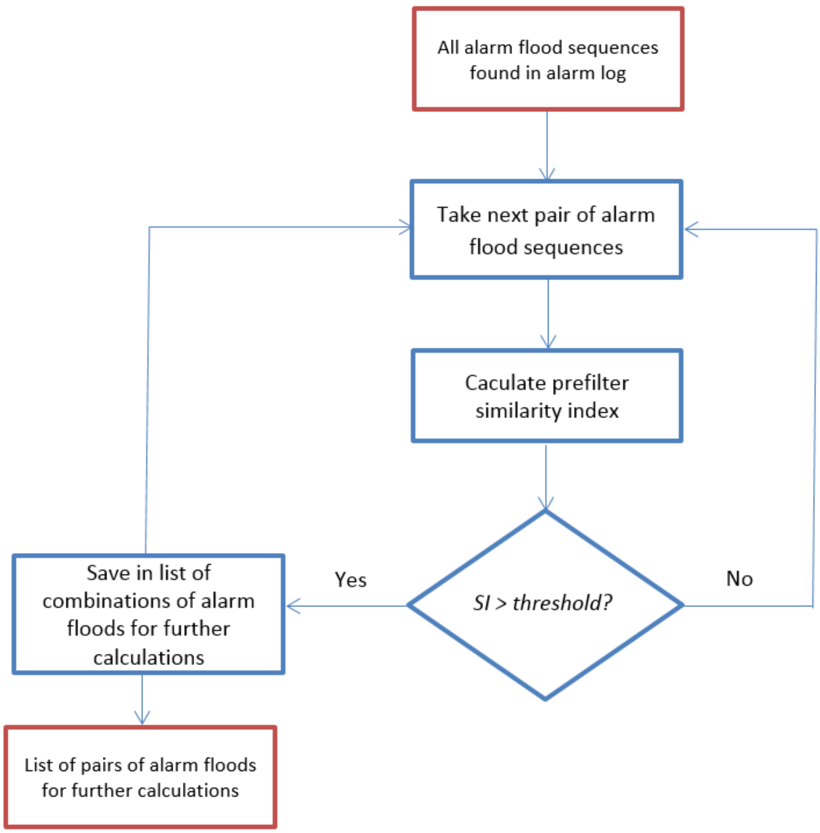


Figure 5.4 Pre-filtering workflow

5.3.2 Similarity Matrix

After filtering out the combinations of alarm sequences that are not similar enough, the MSW similarity index is computed for the remaining combinations and a similarity matrix is built. The similarity matrix is an $N \times N$ symmetric matrix (N being the number of alarm sequences). It contains the similarity indices for all combinations of alarm sequences. The value of an element $[i, j]$ in the matrix is the similarity index computed between the alarm sequence i and the alarm sequence j.

The MSW similarity index of the dismissed combinations is assumed to be a low number (zero was used for the case study). The diagonal entries of the matrix should have a value of one since the similarity index between a sequence with itself is one. However, in order to avoid the clustering algorithm to cluster a sequence with itself the values in the diagonal are forced to be zero.

In Figure 5.5 a portion of a similarity matrix can be seen:

Figure 5.5 Portion of a similarity matrix example

5.3.3 Agglomerative hierarchical clustering

The clustering method used in this thesis is the agglomerative hierarchical clustering. At each iteration, the clustering algorithm accesses the similarity matrix to compute the similarities between a new cluster and the rest of the clusters. The output of the algorithm is a set of clusters where each cluster contains similar alarm sequences. The agglomerative hierarchical clustering algorithm is explained in detail in the section *Agglomerative hierarchical Clustering* in the *Pattern analysis of alarm sequences* chapter.

One of the assumptions of this method is that if a group of alarm flood sequences are similar enough, they are originated from the same process abnormality. Hence, we consider each cluster as a representation of a process abnormality. The alarm flood sequences inside each cluster form a training set from which an "alarm print" of the process abnormality can be extracted. For each process abnormality an alarm template containing the representative alarms that identify this process abnormality can be obtained [Bouchair et al., 2013].

It may happen that while the process abnormality described in one of the clusters occur, other alarms that are not associated to the process abnormality are triggered. These alarms should not be included in the fault template. One way of filtering out these alarms is to consider just those alarms that have a high frequency of occurrence in the training set of alarm floods. If an alarm occurs with a high ratio in the different alarm flood sequences of a given cluster, this alarm is included in the template. The percentage used for the industrial case study described in this thesis is set to 50%, i.e. if an alarm is triggered in more than 50% of the sequences belonging to a given cluster, this alarm will be further considered.

The output of this step is a set of templates defining the process abnormality corresponding to each of the alarm flood clusters. The alarms contained in these templates will be the ones considered in the following steps of the method.

5.4 Mapping

In the proposed approach three process information sources are combined: alarm log, process data and plant topology. Even though the sources differ in the nature of the data contained, they are related to each other. Signals usually measure a property of an asset or the product contained in it. For example a signal generated by a pressure indicator informs about the pressure of the gas contained in a tank. On the other hand, alarms can be assigned to signals (limit alarms) and also directly to assets (e.g. a logic alarm indicating a malfunction of a specific asset). This relation between information sources forms the core of the proposed approach. Hence, being able to link alarms, signals and assets is crucial.

There is no universal naming convention for alarm, signal and assets' tags. This hinders a direct mapping between the information sources. In spite of the lack of tagging conventions, it is usually possible to identify the connections between alarms, signals and assets by the similarity of the symbols contained in the tag names. For instance, the tag name of a temperature controller from section 01 of the case study is TC11 in the P&ID, this controller has a signal associated whose tag is TC11CO.XAY and an alarm with name TC11CO is assigned to this signal, see Table 5.1.

Table 5.1

P&ID asset tag	Signal tag	Alarm tag
TC11	TC11CO.XAY	TC11CO

A way to automatically map the alarm, signals and assets is to use a similarity index as it has been done to cluster alarm flood sequences. However in this case the problem is more simple since there is no time consideration. The selected method to map tags is the original Smith Waterman algorithm [Smith and Waterman, 1981].

Given a list of alarm tag names and a list of signal tag names, the similarity index of each combination of strings (formed by signal, alarm or asset names) is computed. With these indexes a similarity matrix $N \times M$ is built, N being the number of alarm tags and M the number of signal tags. Then a similarity threshold is set, 0.5 was the chosen threshold for the industrial case study. Each alarm tag will be mapped to the more similar signal tag if their similarity index is higher than the threshold value. The same procedure is applied for signal and assets tags and for alarm and asset tags.

After this step, the connection between the different information sources is set. The corresponding process and topology data is available to perform root-cause analysis to isolate the alarm associated to the asset where the fault was originated.

5.5 Root-cause analysis

In the alarm analysis stage, faults that start alarm floods are identified and characterized. However, the causal alarm cannot be isolated relying on the alarms' time-stamps. The first alarm in time cannot be considered the causal alarm (alarm associated to the root-cause). This is due to the fact that the time at which an alarm triggers is highly dependent on the alarm limit settings. Moreover, the time between an abnormality occuring and the corresponding alarm triggering is not a deterministic variable, but a probabilistic one. This is the reason why in order to identify the first alarm of a fault, the process data associated to these alarms is analysed instead. With the use of process data analysis, the precedence relationships between signals can be captured.

The root-cause analysis is performed for each of the alarm clusters found. The signals associated to the alarms within the fault template are selected for the study. Based on the assumption that alarm flood sequences within the same cluster have the same root-cause, the data-driven analysis just needs to be done for one of the alarm flood periods within each cluster.

The methods available in the PDA software tool will be used for this purpose (see Chapter 4 for more details on the methods used in PDA). The result of the process data analysis is a set of suggested signal root-causes. However, there is generally just one root for a specific fault. In order to reduce the number of spurious results from the signal analysis, the results are validated with the information about plant connectivity contained in a graph generated from the intelligent P&ID

[Thambirajah et al., 2009].

The first step for the topology based validation of the data-driven analysis is to identify the assets that correspond to the signals used in the data-driven analysis. These assets will be indicators and controllers. Additionally, the alarm template that is being analyzed there may contain alarms that are not associated to signals but to assets, e.g. alarms indicating a malfunctioning pump. These assets are also automatically included in the topology analysis.

The connectivity of the plant is explored using a graph-search algorithm on the graph capturing the connections between assets in the plant. The depth first search is used for this purpose in the software tool implemented in this thesis. The controllers or indicators suggested as root-causes from PDA are chosen as starting points of the disturbance. The rest of assets selected for the topology analysis are chosen as secondary disturbed elements. Then all feasible paths from the suggested root-cause to each of their secondary disturbed assets are explored. If there are not feasible paths from the root-cause suggestion to several secondary disturbed elements, the root-cause hypothesis is not valid since according to the topology model there is no way that the irregularity in this asset could spread to the other assets.

Once the spurious results of the data-driven analysis have been filtered out and one of the root-cause suggestions has been validated with plant connectivity. The first alarm under which the other alarms within the template will be grouped is the alarm associated to the root-cause asset.

5.6 Plant areas selection

Experimental analysis of industrial alarm records showed the existence of alarm floods caused by alarms notifying controllers put on manual mode or alarm floods reporting malfunctioning ports when an electronic device failed. This kind of alarm floods are out of the scope of this thesis since they are not caused by a disturbance or an asset fault propagating through the process via the material, energy or information connections. Finding the first alarm with the presented method would be impossible since there are no signals associated to these alarms.

For some of the plant areas these type of alarm floods are frequent. Filtering out these areas is beneficial mainly for two reasons:

- Plant-wide monitoring is a large scale monitoring problems. The alarm management system of these plants usually have several thousands of unique alarms configured to control the process. It is not rare that around ten thousand alarms trigger each month. The computation time needed to calculate all similarity indexes between all alarm flood sequences found in the alarm logs can easily be longer than one day on a single computer. If those plant areas containing alarm floods caused by logical sequences are removed from the analysis, a reduction of the problem complexity can be achieved.

- Including plant areas with alarm floods not caused by a disturbance propagating through the plant gives rise to a big portion of alarm clusters that are not in the scope of this method. In this situation, the user has to examine and discard these clusters manually, which would be a tedious and time consuming task.

A solution to identify the "interesting" areas of the plant is to first perform the first three steps of the method (alarm log analysis stage) using as an input just those alarms associated to signals (limit alarms). The reason behind this is that alarm floods due to propagating disturbances have a big proportion of limit alarms. Therefore, if areas of the plant with similar periods of high number of limit alarms are selected, clusters of alarm floods caused by a disturbance propagating through the plant are targeted.

The purpose of the proposed method is to cluster all alarms originating from the same process abnormality under the causal alarm. In order to include in this cluster also alarms that are not associated to signals, the alarm analysis stage is repeated including all alarm occurrences of the selected areas. All consequential alarms are therefore identified independently of their type.

The pre-analysis of the alarm log described in this section reduces the computational time of the similarity matrix and saves the user the tedious task of selecting the appropriate clusters when dealing with large alarm records.

6

Software tool

A substantial part of the work reported in this thesis involves implementing a software tool that guides the user through the steps of the proposed method. The tool was developed under the .NET framework using the development tool Microsoft Visual Studio 2013. The application is implemented in C#, a high level programming language.

The application consists of a main form that contains three user controls: Alarm Log Analyzer, Process Data Analyzer and Topology Analyzer. The first user control performs an alarm log analysis. The second carries out a process data root-cause analysis. The last one validates the results obtained from the process data analysis. When starting the application, only the Alarm Log Analyzer is available for the user. Once the alarm analysis is completed, the Process Data Analyzer is enabled. And once the signal analysis is completed, the Topology Analyzer is enabled.

In this chapter a detailed description of the functionalities of the developed software tool is presented.

6.1 Alarm log analysis

When the application is run, the alarm analysis window is shown (see Figure 6.1).

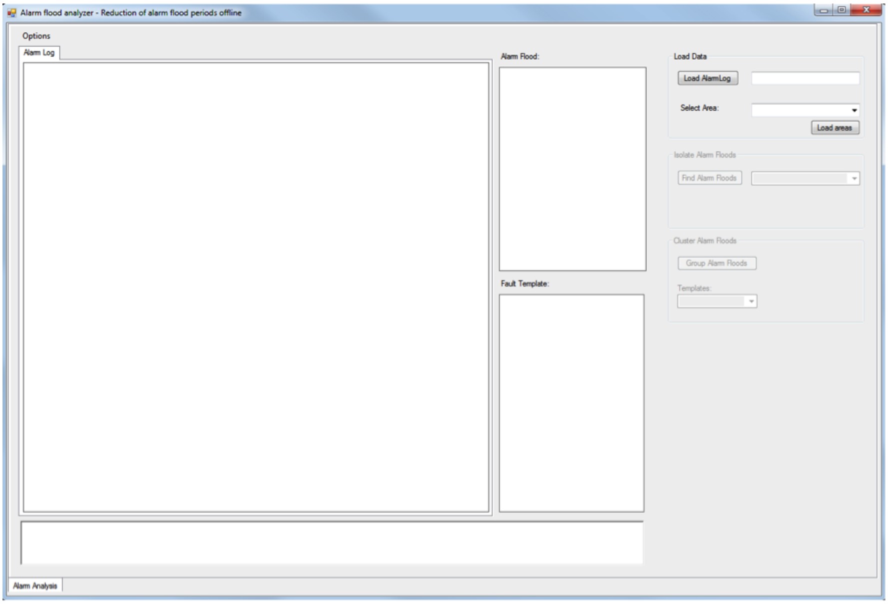


Figure 6.1 Start screen, alarm analyzer

6.1.1 Parameter selection

By clicking on the options menu and then on "Change Parameters", the parameters of the alarm analysis can be defined (see Figure 6.2).

Alarm flood identification:

- Interval length: time length of the intervals in which the alarm log is divided (in min).
- Number of alarms: number of alarm occurrences within an interval to consider these intervals an alarm flood episode.

Alarm flood grouping parameters:

- Prefilter threshold: number between 0 and 1 representing the prefiltering threshold. The larger the value the more combinations of alarm floods will be filtered out in the prefiltering step.
- Clustering threshold: number between 0 and 1. The larger the value the more similar two alarm floods have to be in order to group them under the same cluster.
- Sigma: represents the time delay allowed between alarm occurrences within different alarm sequences to be considered as a match by the algorithms.

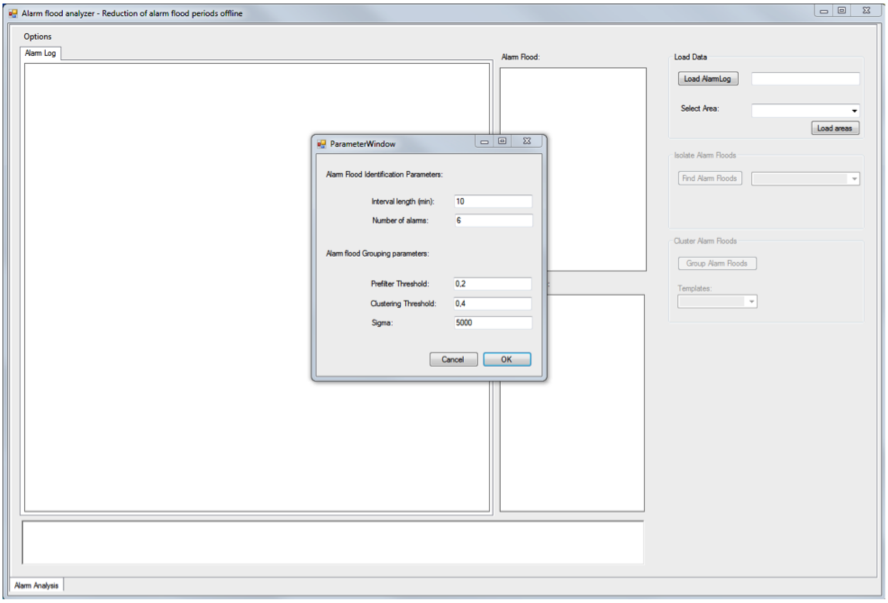


Figure 6.2 Parameter selection window

6.1.2 Data loading

The next step is to load the alarm log. When the button “Load AlarmLog” is pressed an open file dialog is displayed. An alarm log saved in a compact proprietary format (.pca) can be chosen from the corresponding directory.

Once the alarm log is loaded, a frequency plot showing the number of alarms triggering within each time interval of the alarm log is displayed. The X axis of the plots in Figure 6.3 represents time, and is divided into intervals of the chosen length. The Y axis shows the number of alarm occurrences within each of these intervals, refer Figure 6.3.

The alarm log can be filtered by plant areas. If the focus of the analysis is on specific plant areas, they can be chosen on the ComboBox with the label “Select Area”. Once this is done and the button “Load area” is pressed, the frequency plot is updated according to the user’s selection. In the example area 01 is selected, displayed in Figure 6.4.

6.1.3 Alarm flood periods isolation and alarm sequences construction

By pressing the button “Find Alarm Floods”, the periods considered as alarm floods according to the user’s parameter choice are highlighted in red on the frequency plot.

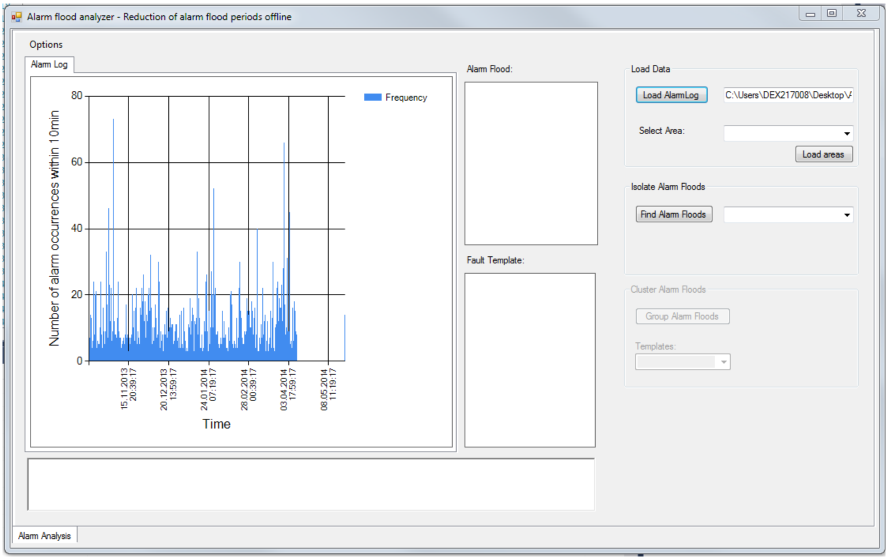


Figure 6.3 Alarm log display

The alarm flood sequences are constructed and can be displayed on the ListView labeled as “Alarm flood:” by selecting the desired sequence on the ComboBox next to the “Find Alarm Floods” button, see Figure 6.4.

6.1.4 Clustering of alarm flood sequences

Once the alarm flood periods are isolated, the “Cluster Alarm Floods” groupBox is enabled. By pressing the “Group Alarm Floods” button, the clustering step is started. The clustering process may take some minutes depending on the number and the length of the alarm flood sequences found in step 2, for more information about the clustering method refer to [Master Thesis report]. When the clustering is completed a new tab is added on the top of the window, called “Alarm Clustering”. Here, the different clusters found and correspondinglythe alarm sequences within each cluster are displayed (Figure 6.5).

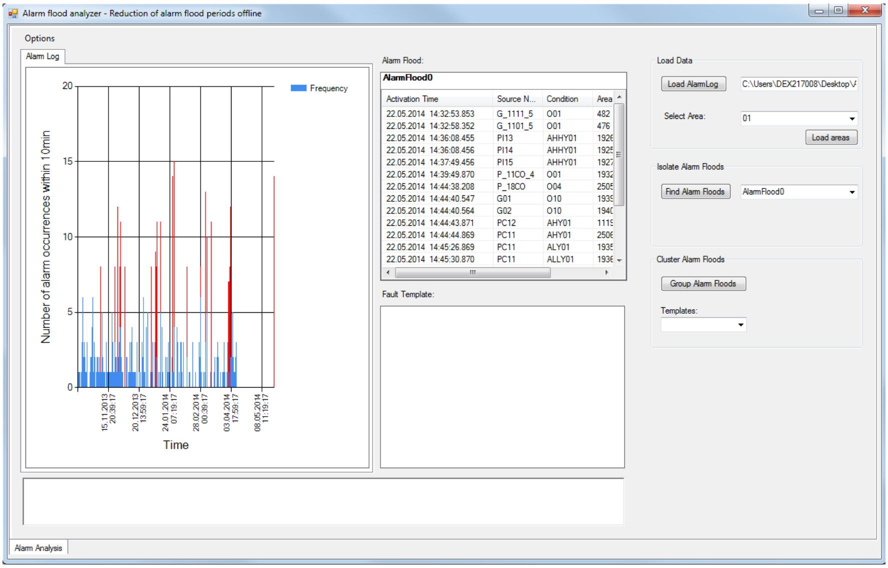


Figure 6.4 Alarm flood periods isolated

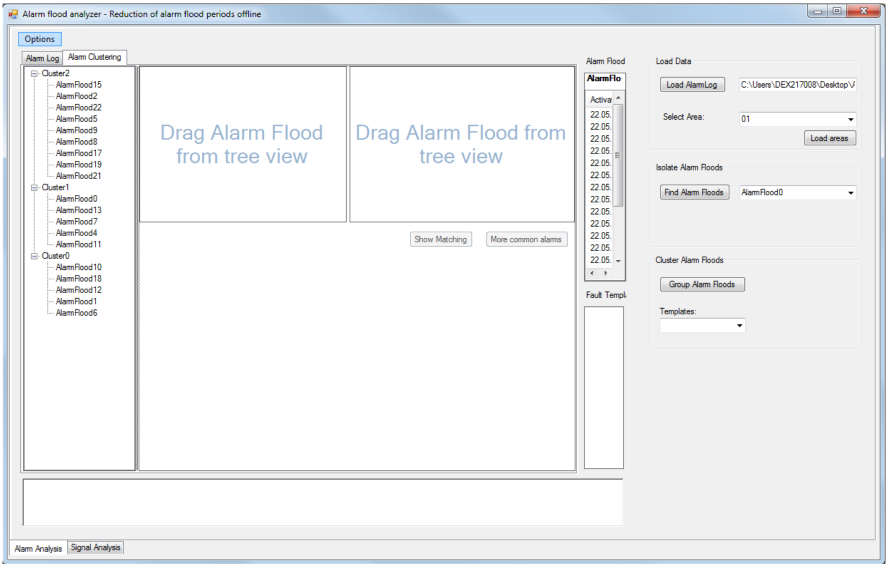


Figure 6.5 Alarm flood sequences clustered

The alarm sequences within the treeView on the left side of the screen (Figure

6.5) can be displayed by dragging and dropping in the two visualization windows labeled “Drag Alarm Flood from tree view”. Once two alarm sequences are dragged into these windows, the matching alarms found by the Modified Smith-Waterman algorithm can be seen by pressing the button “Show Matching” (Figure 6.6). Below the two windows, a chart is displayed. On this chart the matching alarms are displayed over the time. Time zero is taken as the time where the first matching alarms triggered (in yellow in the example displayed in Figure 6.6). The colors of the points correspond to the colors of the alarm occurrences displayed on the top windows. The points above the X axis are the matching alarms of the alarm flood sequence displayed on the left window. The points below the X axis are the matching alarm occurrences of the alarm flood sequence displayed on the right window. For instance, in Figure 6.6, alarm PC12/ALY01 which is the first matching alarm occurrence. It is highlighted in yellow and appears in the chart at time 0 for both sequences (yellow square and yellow circle). Alarm PI13_1/AHHY01 which is the third alarm matching found, it is identified by a dark green color. On the plot corresponding to the first sequence (above X axis) a dark green square is plotted at time 228s since this alarm occurred 228s after alarm PC12/ALY01 in the first sequence. On the plot corresponding to the second sequence (below the Y axis) a dark green circle is plotted at time 253s since for the second sequence this alarm triggered 253s after the reference alarm (PC12/ALY01).

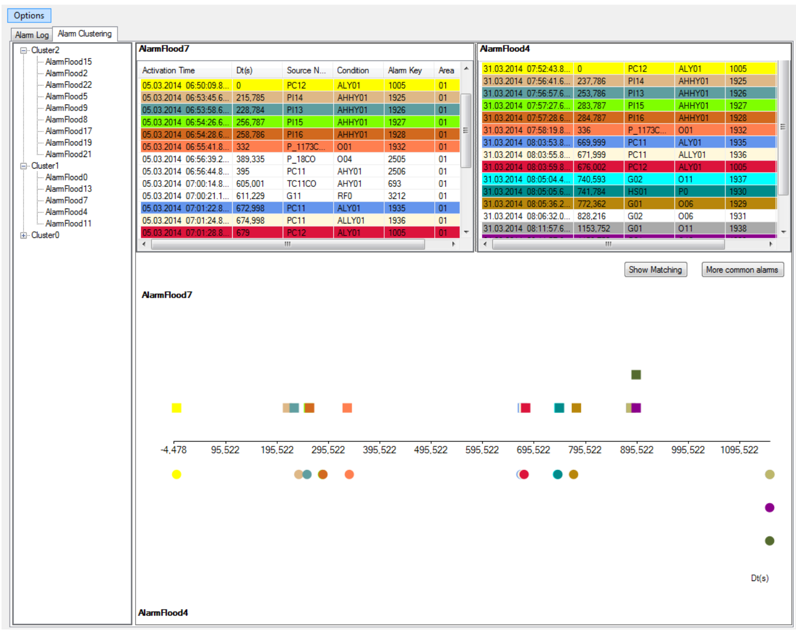


Figure 6.6 Alarm sequence matching visualization

By using this chart, the user can analyze, at a glance, the similarity between two alarm flood sequences.

The hypothesis of the method is that if two alarm flood sequences are similar enough, then they are caused by the same fault. Hence, each cluster represents a different fault. For each cluster an "alarm template", i.e. alist of alarm types that characterize the fault, is built. This template can be seen by selecting the corresponding cluster on the ComboBox under the label "Templates:", refer Figure 6.7.

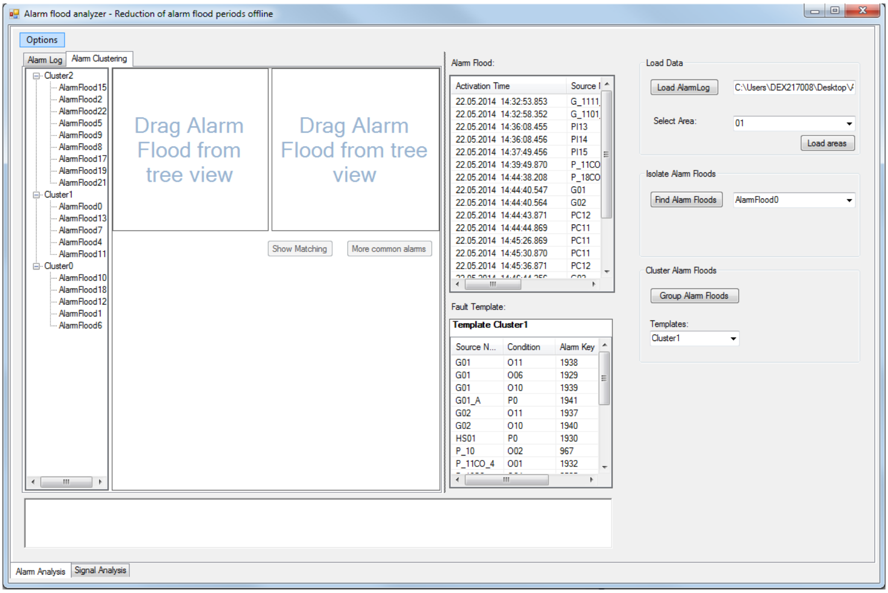


Figure 6.7 Template visualization

When the alarm flood clustering step is finished, the signal analysis tab (see bottom left Figure 6.7) is created.

6.2 Process data analysis

6.2.1 Data loading

The Signal analysis window is displayed in Figure 6.8 below.

Before starting the process data analysis, alarm and signals are mapped. The process data are first loaded. An open file dialog is displayed when the button "..." is pressed. Once the excel file that contains the process data is selected, the corresponding Excel ranges of the tag names, time stamps and signal values are written

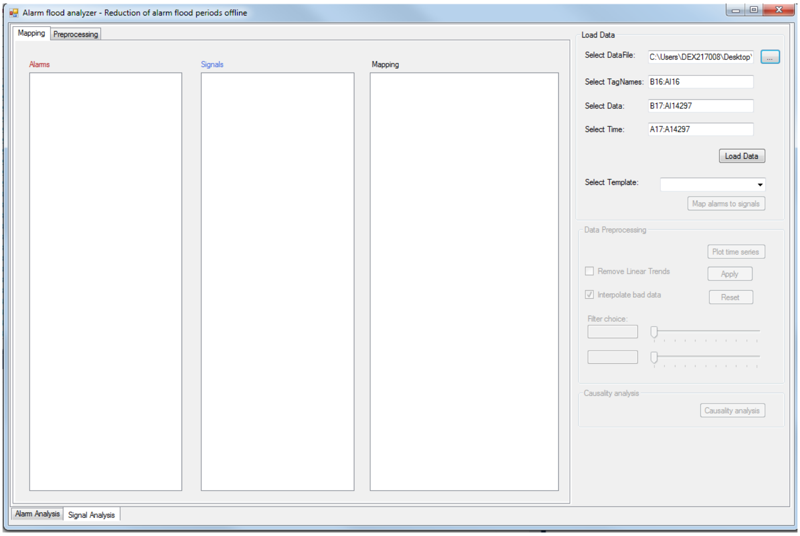


Figure 6.8 Process data analysis window

in their corresponding text boxes. The button “Load Data” is clicked, the dataset is loaded and the signal tags are displayed on the TreeView labeled as “Signals” (in blue in Figure 6.9).

The template that to be analized is chosen by selecting the template name on the ComboBox labeled “Select Template:”. The template tags are displayed on the TreeView labeled as “Alarms” (in red in Figure 6.10). The user is able now to map the alarm tag names to the signal tag names. Additionally, the user can select which tags are going to be considered in the mapping by checking/unchecking the box on the left of each tag name.

6.2.2 Mapping

Once the “Map alarms to signals” button is pressed, the results of the mapping are shown on the TreeView labeled “Mapping”, Figure 6.10. The Smith-Waterman algorithm is used to extract a similarity index between the different tags’ names. Each alarm tag strings is linked to the signal tag string for which the maximum similarity index is obtained.

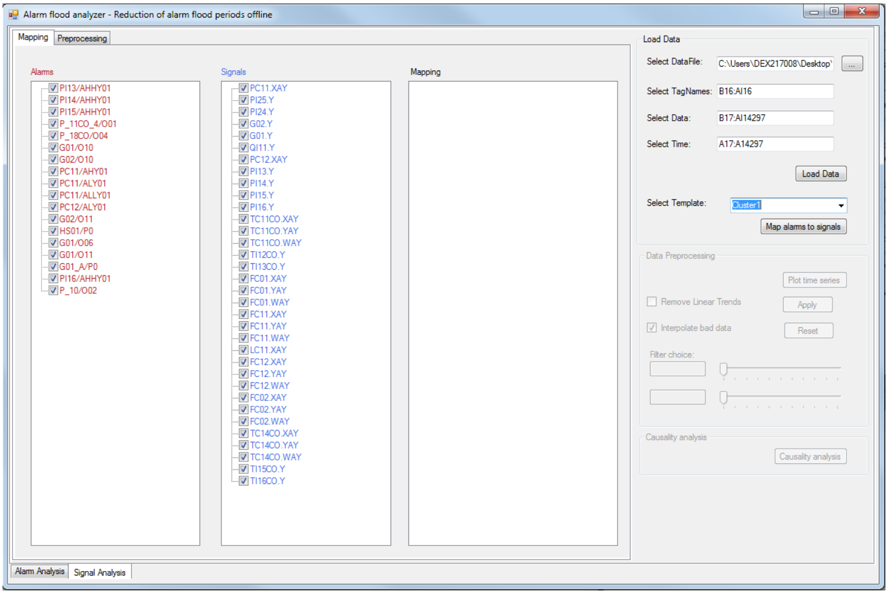


Figure 6.9 Signal data loaded

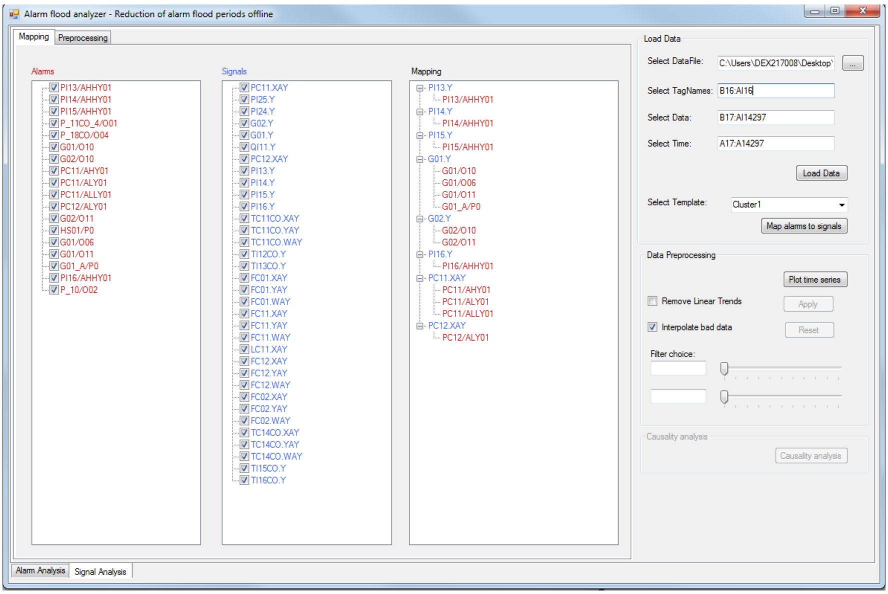


Figure 6.10 Mapping results

The names corresponding to alarms are displayed in red, while the tags corresponding to signals are displayed in blue. On the mapping TreeView, under each signal tag, those alarms that have similar names are displayed.

The user can manually adjust refine the mapping results. A node within the tree can be removed by selecting it and pressing the key “Delete”. New nodes can be added by dragging and dropping nodes from the signals and alarms TreeView to the corresponding position on the mapping TreeView. The user must notice that for obtaining valid results the structure originally showed must be kept, i.e., signals are parent nodes and alarms are child nodes, the depth of the tree should not be larger than two.

The signals displayed on the mapping view are used for following the process data analysis step.

6.2.3 Preprocessing

Here preprocessing of the signal data can be performed. By pressing the button “Plot time series” the time trends (Figure 6.11 top) and spectrum (Figure 6.11 bottom) of the signal tags in the mapping TreeView are plotted on the “Preprocessing” TabPage. The time interval to be considered in the analysis can be chosen using the trackbars below the plots. The filtering interval is defined with the trackbars on the right (or by entrying the filter settings on the corresponding textbox and pressing “Enter”). Once the settings are defined, the data preprocessing is performed by clicking on the button “Apply” (see Figure 6.11). Results of the preprocessing are displayed in Figure 6.12.

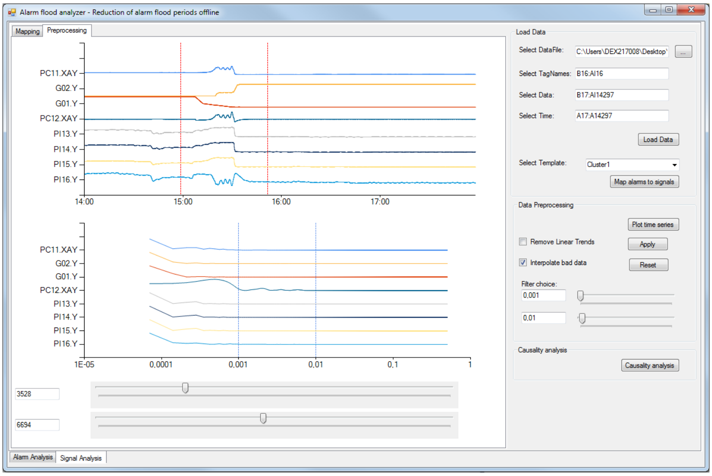


Figure 6.11 Signal preprocessing

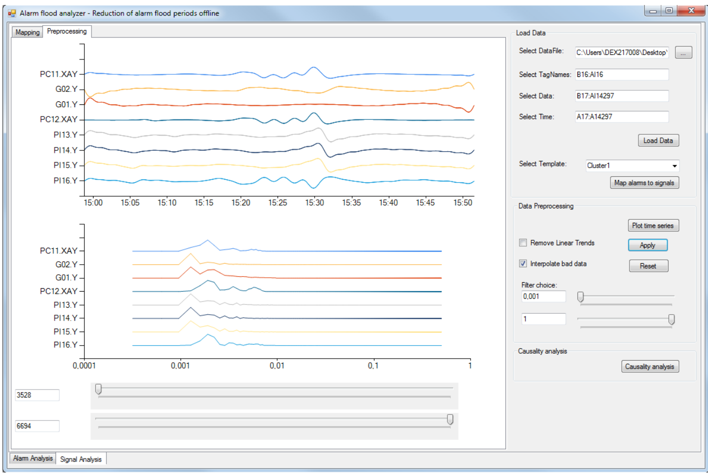


Figure 6.12 Signal preprocessing results

The button “Reset” can be pressed in order to undo the data changes done.

6.2.4 Causality analysis and results

The causality analysis is launched by pressing the “Causality analysis” button. A method based on transfer entropy is implemented in this tool. Transfer entropy is a information based method evaluating the predictability of a variable from a second variable. The results of the computation are plotted and displayed in the tab named "Process data analysis result" (Figure 6.13 top).

The results are a bubble chart representing transfer entropy based causality matrix (left side) and a list of alarms associated to the analyzed signals, these are the alarm tags that are grouped under the causal alarm, refer Figure 6.13.

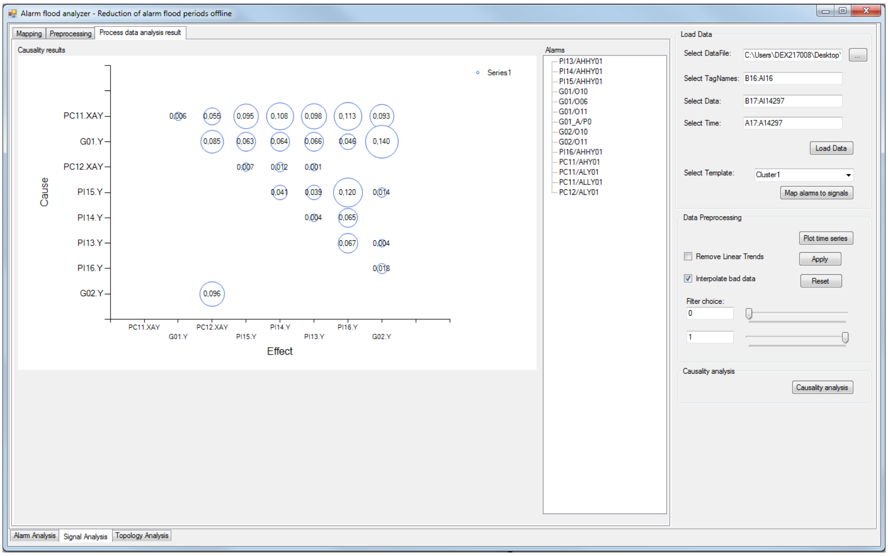


Figure 6.13 Causality results

For the example shown in Figure 6.13, the size of the bubble at the intersection of tag PC11.XAY on the vertical axis labeled "Cause" and the tag GI01.Y on the horizontal axis labeled "Effect" is to be interpreted as the extent to which PC11.XAY influences GI01.Y and this is quantified by the causality measure shown in the center of the bubble and calculated from the transfer entropy of the time trends of the two tags. The algorithm used to sort the cluster's tags places the tag suggested as a root-cause of the disturbance at the top of the plot. In this case the suggested root cause is the signal PC11.XAY.

Once the causality calculations are finished, the topology analysis tab (bottom left Figure 6.13) is created. The root-cause suggestion obtained by the signal analysis can be validated using the process topology model.

6.3 Topology analysis

6.3.1 Data loading

The Topology analysis window is displayed in Figure 6.14.

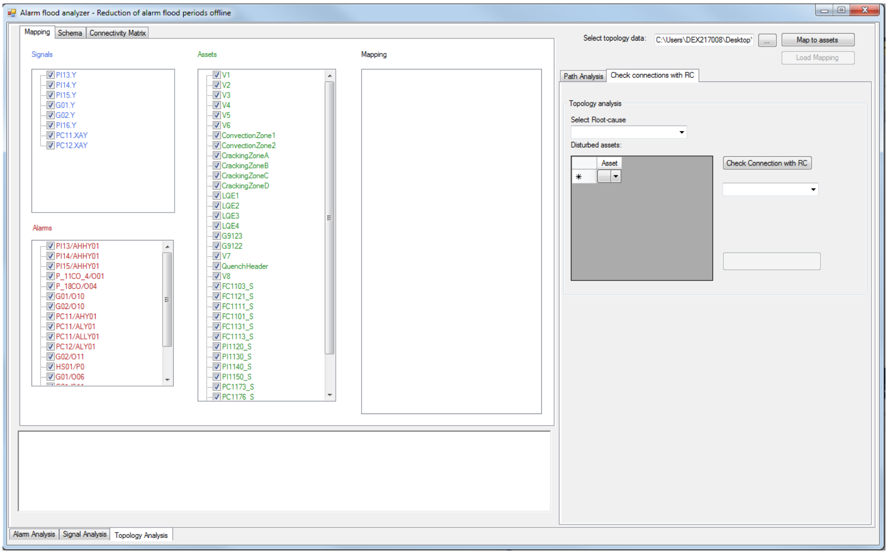


Figure 6.14 Topology analysis window

Once the topology analysis window is loaded, the signals TreeView (blue) and the alarm TreeView (red) are automatically filled with the signals considered in the process data analysis step and the alarm tags within the template being studied.

By clicking on the button "…" an open file dialog is displayed. Here, the user can choose the CAEX file containing the topology model of the plant under study. When the topology model is loaded, all asset tags found in the CAEX file are displayed on the assets TreeView (green, Figure 6.14 middle). The plant schematic is loaded in the tab "Schema" (top left Figure 6.14) and the connectivity matrix is loaded on the tab "Connectivity Matrix", see Figure 6.14.

6.3.2 Mapping

Before starting the topology validation, the assets contained in the topology CAEX file must be mapped to the signals and the alarms. The user can choose which tags to consider for the mapping by checking or unchecking the box on the left of each tag. By pressing the "Map to assets" button, alarms and signal tags are linked to assets. First, those assets associated to the signals used in the signal analysis are found. Then the alarms connected to the signals are identified and grouped under the corresponding asset (this information is extracted from the mapping of the Process data analysis window, see section 6.2.2). Finally, the algorithm identifies assets that are assigned directly to alarms (not via a signal), for instance an alarm notifying that a pump is malfunctioning.

The mapping results are displayed on the mapping TreeView, Figure 6.15. Green color corresponds to asset tags, red color to alarms and blue color to signals.

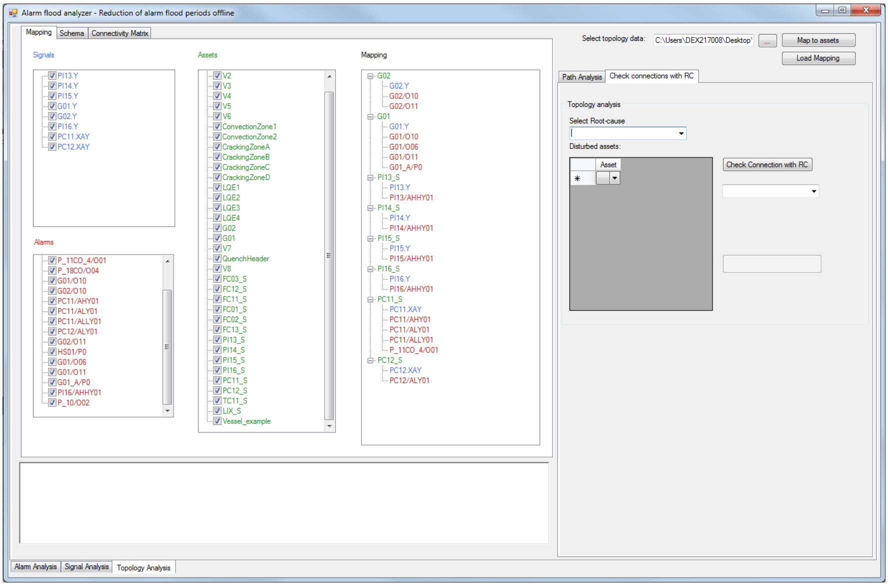


Figure 6.15 Mapping results

For example in Figure 6.15 the signal GI02.Y is assigned to the asset G02. The alarms that were before mapped to this signal can be found under the asset as child nodes.

The user can add nodes and child nodes by dragging and dropping the nodes from the signals, alarms and assets TreeViews to the mapping TreeView. The user can also delete a node from the mapping TreeView by selecting a node and pressing the key “Delete”.

Afterwards, the user can press de button “Load Mapping” (Figure 6.15) to load the assets in the topology analysis. The ComboBox under the label “Select Root-cause" is filled with the asset connected to the signal suggested as a root-cause by the process data analysis. The DataGridView under the label "Disturbed assets" is filled with all the assets present in the mapping TreeView that are not linked to the suggested signal root-cause, these are the secondary disturbed assets (Figure 6.16).

6.3.3 Topology validation

There are two options to validate the results from PDA using the topology analyzer:

- "Check connections with RC" (root-cause).
- "Path Analysis" (not used in the method presented in this thesis).

Each option can be found under its corresponding tab labeled.

Check connections with the root-cause This option allows the user to check if there exist feasible paths connecting the root-cause with all the secondary disturbed elements. If this is the case, the text "Feasible" is shown inside the result bottom and the button turns green. Hence, the suggested root-cause is validated, and the alarm associated to the root-cause asset is the suggested causal alarm. All other alarms contained in the mapping TreeViw are grouped under this causal alarm.

Additionally, the paths found are plotted on the process schematic (Figure 6.17). A graph is also displayed below the Topology analysis GroupBox in order to show the order in which the disturbance affected the assets (since this cannot be seen on the process schematic). The root-cause asset is represented as a yellow circle. The circles corresponding to the secondary disturbed elements are colored with their respective colors as displayed on the Disturbed assets DataGridView. Assets within the path which are not in the list of secondary disturbed elements are represented by a light blue circle. Finally, sensors that are in the path but were not suggested by the signal and alarm log analyses as disturbed elements are displayed in red. These assets shoucl be considered carefully as they might be not working properly and a maintencance operation on them might be required. An example of the topology validation is shown in Figure 6.16 and 6.17.

If there are not feasible paths connecting the root-cause with one or more of the secondary disturbed elements the text "Not feasible" is displayed to the user in the result button and the button turns red. The feasible paths are displayed on the graph. The tool notifies which asset is the one not connected to the root-cause suggested on the TextBox at the bottom of the window.

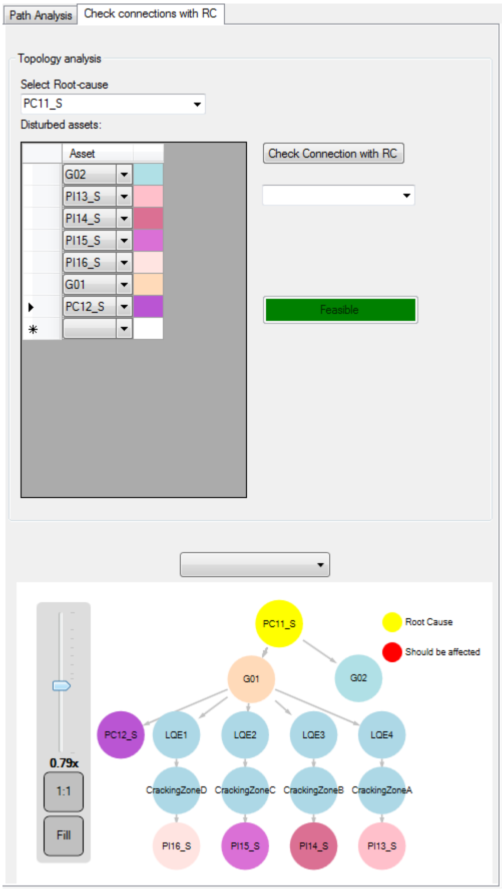


Figure 6.16 Root-cause validation using process topology

In the example displayed in Figure 6.16, the disturbance originates on the controller PC11_S, and it spreads to the valves G01 and G02. From G01 the disturbance takes five parallel paths affecting PC12_S, PI15_S, PI15_S, PI14_S and PI13_S.

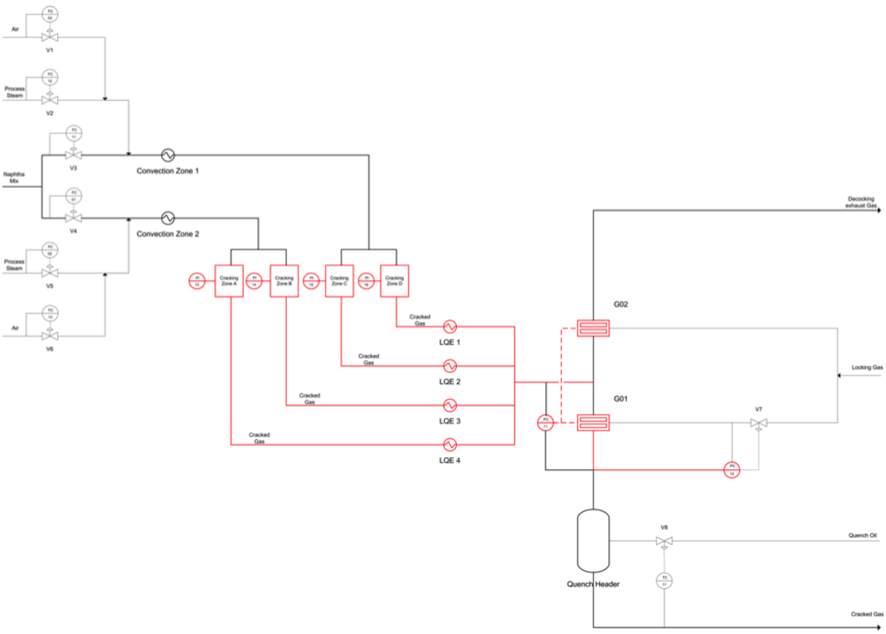


Figure 6.17 Schema with the path highlighted

7

Industrial case study

This chapter illustrates the benefits of the method by applying the developed workflow to real data originated from an ethylene plant. Due to a confidentiality agreement information regarding the process and the plant has been protected in this discussion.

The case study illustrates the advantages of using a combined information source (alarm, process and topology data) from a real process plant. The case study presented is a large scale monitoring problem. The alarm management system of the plant has more than 3800 unique alarms configured to control the process. The analysis is performed on 6 months of alarm data, during which more than 48000 alarms triggered.

7.1 Process description

The method is applied to an ethylene plant. In this plant a steam cracking operation, i.e., a pyrolysis process in which saturated hydrocarbons are broken down into smaller carbon chains. The feed material is naphtha, i.e., a mixture of C_5 to C_{12} hydrocarbon molecules. The naphtha is diluted with steam and driven inside metal tubes in absence of air through a furnace where is heated. The gaseous mixture of naphtha and steam is quickly passed through the furnace (in less than a second) at low pressure in order to prevent its cracking to carbon form.

The gas is then quickly quenched, to avoid any further reactions. First in a line heat exchanger and then in a quenching header using quench oil. Subsequently, the cracked gas is separated. Besides of ethylene, the following products can be produced in the cracking plant:

- Propylene
- Hydrogen

- Methane
- Gasoline
- Acetylene
- Aromatics
- C9+product

An schematic of the process is shown in Figure 7.1.

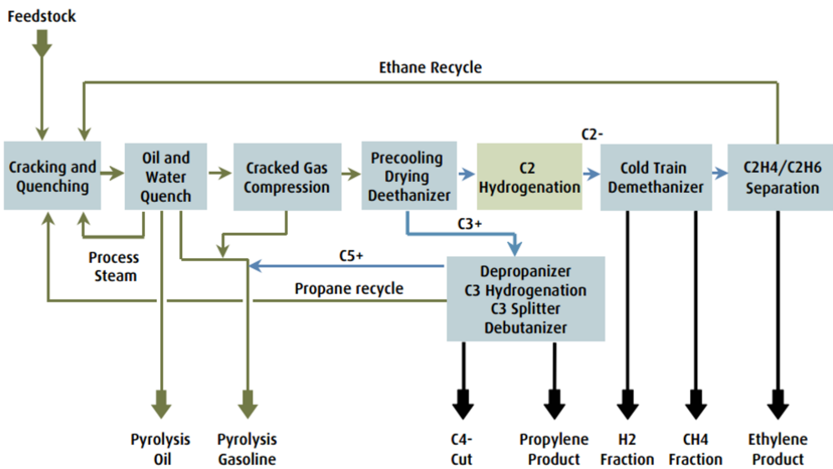


Figure 7.1 Cracking plant schema

7.2 Plant area identification

As stated before in Chapter 5, an identification of plant areas with alarm floods caused by a disturbance propagating through the plant is performed. To do so, the first step of the method is applied to limit alarm logs. The reason is that alarm floods produced by a propagating fault contain a big proportion of limit alarms.

The criteria to select the areas for the case study are the following:

- Relatively high number of periods with large amount of limit alarms (>5 in the 6 months log).
- Some similarity between those periods.

The required parameters for the alarm analysis are listed in Figure 7.2 together with their settings.

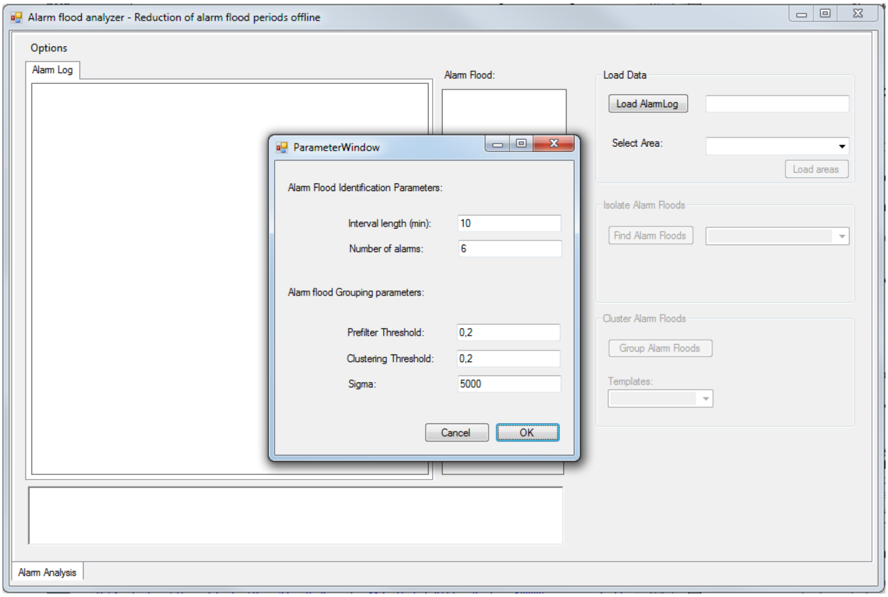


Figure 7.2 Parameter selection

The time interval granularity of the alarm log is set to 10min since the definition of an alarm flood is a time interval containing more than 10 alarms per 10 min. The required number of alarms is set to six. The choice of six alarms is due to the fact that only limit alarms are taken into account. If a period of 10 min has six limit alarm occurrences, it is probable more than 10 alarms would trigger. The pre-filtering threshold is set to 0.2 as well as the clustering threshold. Sigma is 5000 allowing alarms with a time delay of around 3.5 min to be matched.

The used alarm log is in a file with format .pca (“compact proprietary format”) for which those alarm types that are not limit alarms have been previously removed. The software tool automatically removes chattering alarms from the loaded alarm log. Figure 7.3 shows an example of alarm frequency plot in the developed application.

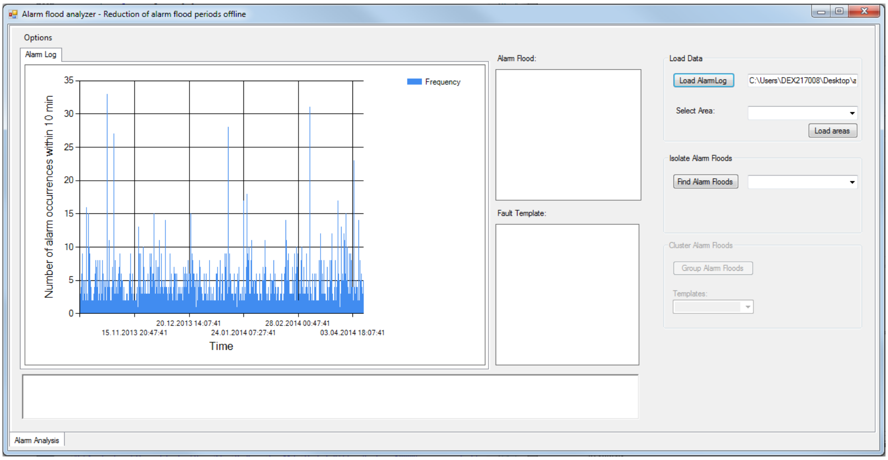


Figure 7.3 Alarm log loaded

The second step of the method is performed by clicking on the “Find Alarm Floods” button. The time periods corresponding to alarm flood episodes according to the parameter choice are highlighted in red in the alarm log frequency plot and the highlighted alarm flood sequences are available for visual inspection by selecting their name in the combobox next to the button pressed, Figure 7.4.

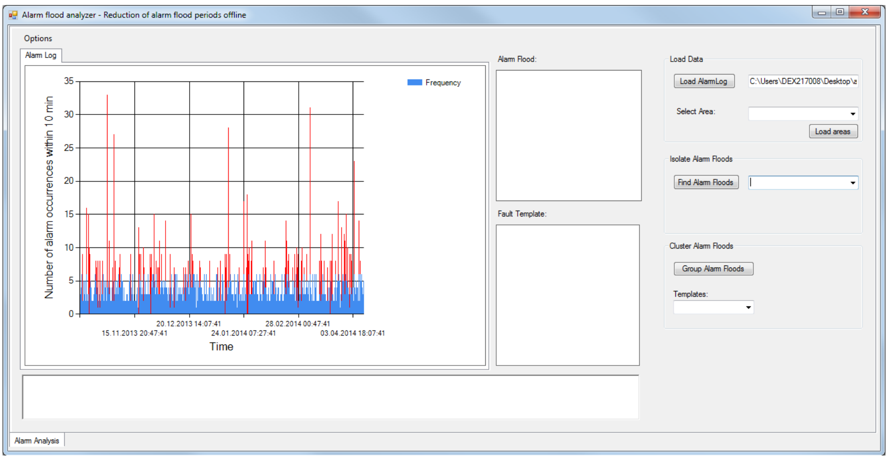


Figure 7.4 Alarm floods identified

By clicking on the “Group Alarm Floods” button (Figure 7.5), the alarm flood sequences are clustered according to their similarity. In the example of Figure 7.5, a total of 20 clusters is obtained. The largest cluster (Cluster 13), containing ten alarm sequences, is further analyzed. By dragging and dropping two random alarm sequences from this cluster to the two windows on the top and by pressing the “Show Matching” button, one can notice that the matching alarms of these periods come from the plant area “01”, see (Figure 7.5).

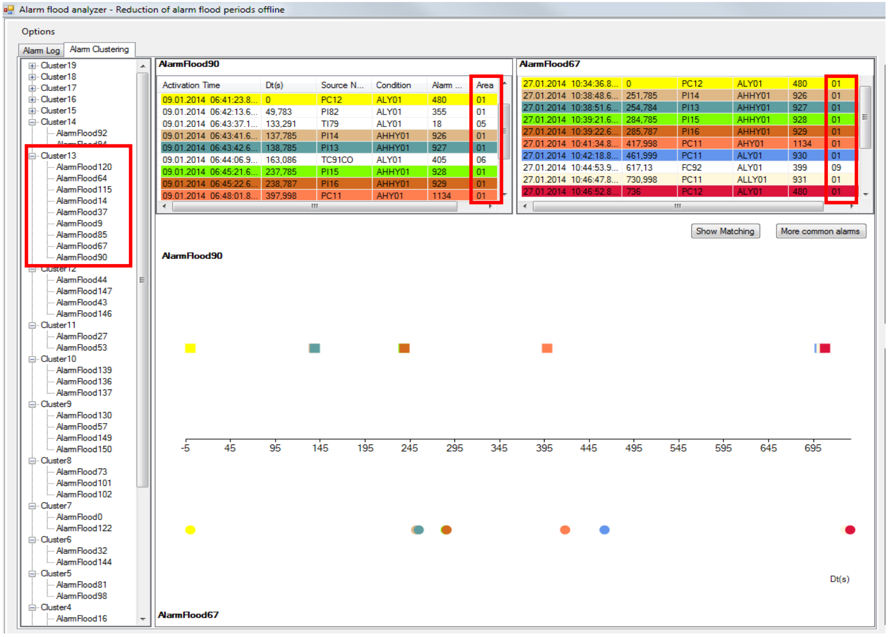


Figure 7.5 Clustering results

According to the chosen criteria area 01, is an area of interest: it has relatively high number of time periods with a large amount of limit alarms (10 periods). Additionally, those periods have some similarity as they belong to the same cluster.

In the following section, plant area 01 is analysed.

7.3 Analysis area 01

7.3.1 Detailed description

Area 01 is the cracking and quenching section of the ethylene plant. A schematic of this area is provided in Figure 7.6.

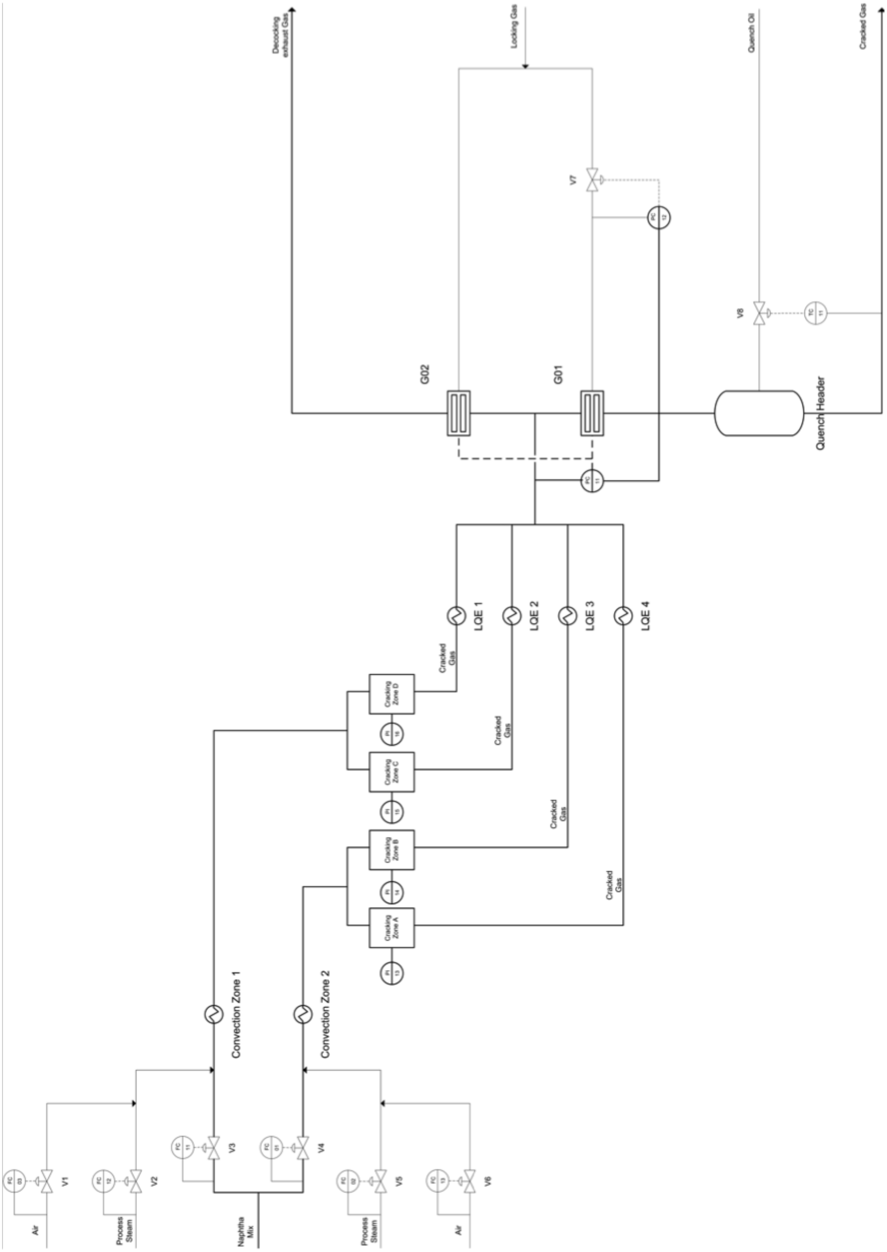


Figure 7.6 Area schematic

During normal operation (cracking mode) valve G02 is fully closed and valve G01 is fully opened. The naphtha mix feed is divided into two lines whose flow is controlled by the controllers FC11 and FC01. The naphtha mix is later mixed with steam whose flow is controlled via flow controllers FC12 and FC02. Each one of the lines containing a mixture of naphtha and steam is divided into two and driven through the coils located in the cracking zones inside the furnace. The naphtha and steam mixture is heated very fast via radiation until it cracks. Following this step, the cracked gas is cooled with linear quench exchangers where the produced high-pressure steam is recycled in the plant. The cracked gas is then further cooled in the quench header by direct contact with quench oil. The flow of quench oil coming into the quench header is controlled via temperature controller TC11. After this, the cracked gas is further treated
.

During cracking, "coke" (a form of solid carbon) is produced. The accumulation of coke in the coils reduces the efficiency of the heat exchange process (see Figure 7.7). Every 60 days the furnace is therefore cleaned by diving a mixture of air and steam in absence of hydrocarbons through the furnace coils. The mixture converts the solid carbon (coke) into carbon monoxide and carbon dioxide. In order to switch the operation mode from cracking mode to decoking mode, the naphtha feed is stopped by closing valve G01 while valve G02 and is opened allowing air feed. Once the coils are cleaned, the furnace is reconnected and the plant is back to the cracking operating mode.

Figure 7.7 Coke accumulation in pipes.

7.3.2 Alarm log analysis

This section describes the alarm log analysis for area 01 of the ethylene plant. The input of this stage is the alarm log of just area 01 and the output is the alarm flood sequences found clustered according their similarity.

The first step in the alarm analysis software tool is to choose the parameters for the alarm analysis. In the plant example the time interval length is set to 10 min and the minimum number of alarm occurrences to consider a time period as an alarm flood episode is six alarms. The formal definition of an alarm flood is a time interval with an alarm rate higher than 10 alarms per 10 min and per operator. In the plant of the case study an operator supervises more than one area. Therefore, it seems reasonable to set the minimum alarm rate to less than 10 alarms per 10 min. Additionally, the value of the clustering threshold is doubled to 0.4. As only alarms originated from area are 01 considered in the analysis, it is expected that there will be less unrelated alarms and therefore the similarity index between sequences is expected to be higher. The parameter settings are given in in Figure 7.8.

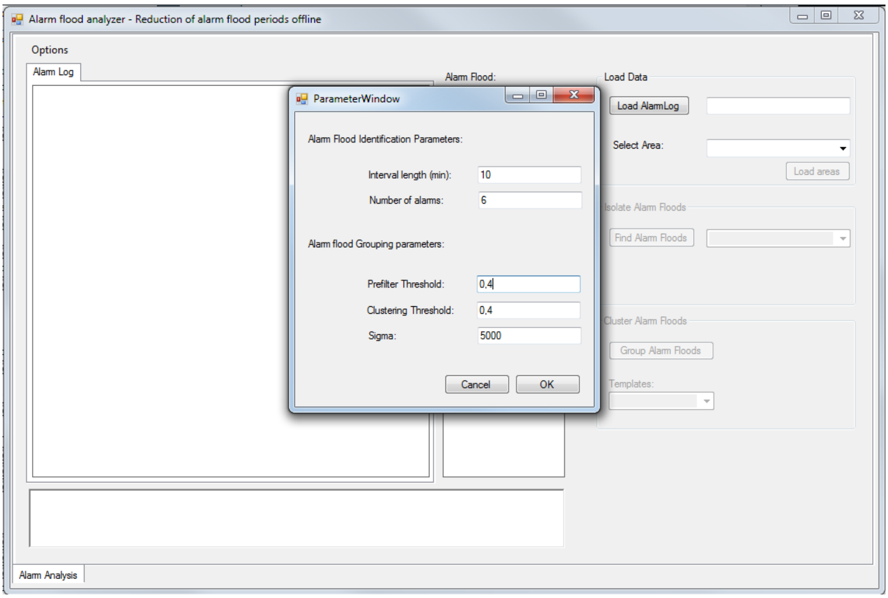


Figure 7.8 Parameter selection

Once the parameters are chosen, the alarm log including all alarm types is loaded. In order to filter out those alarms that do not belong to area 01, the area is chosen in the ComboBox labeled as "Select Area:" and the button "Load areas" is pressed

By pressing the button “Find Alarm Floods”, the time periods with high number of alarms are identified and the alarm flood sequences are constructed. A total of 23 alarm flood periods are found (see Figure 7.9).

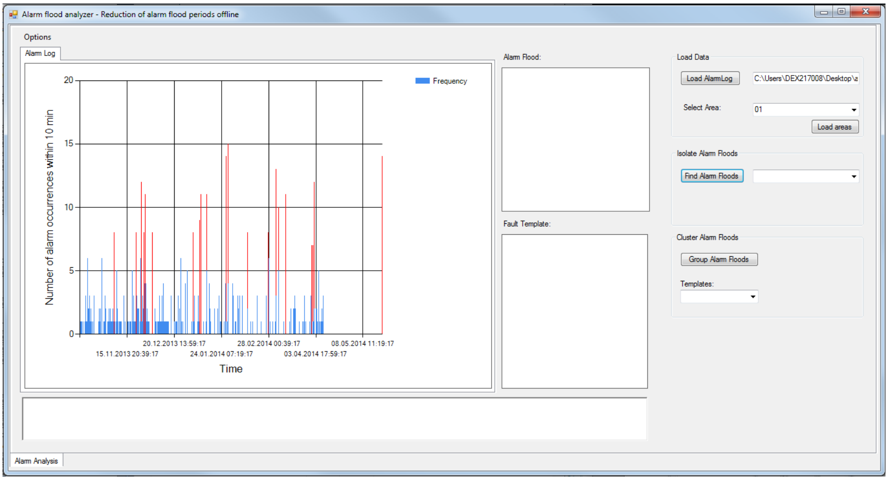


Figure 7.9 Alarm flood periods identified

The alarm flood sequences are clustered according to their similarity by clicking on the “Group Alarm Flood” button. Three clusters are obtained: Cluster 0, Cluster 1 and Cluster 2. Out of the 23 analyzed alarm sequences, four do not belong to any cluster and 19 are clustered under one of the three clusters (see Figure 7.10).

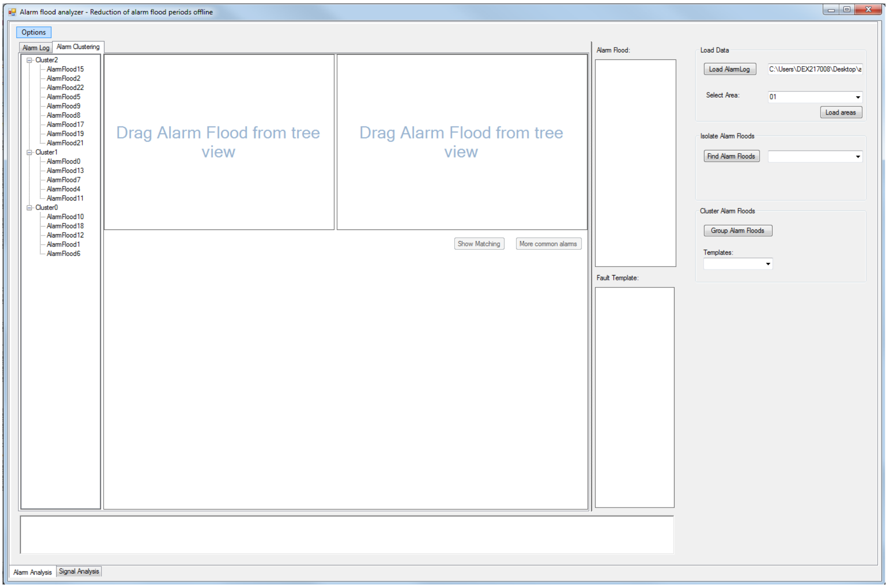


Figure 7.10 Alarm sequences clustered

The alarm sequences within each cluster and their similarities are analyzed by dragging and dropping the alarm sequences from the tree view to the upper windows and by pressing the button “Show Matching”. Figures 7.11, 7.12 and 7.13 show the results obtained when two alarm sequences taken from each one of the three cluster are compared.

In Figure 7.11, two alarm sequences taken from the first cluster (Cluster 0) are compared. The plot in the bottom part indicates that a small portion of the alarms at the beginning of the sequence are triggered and that after approximately 10 min the remaining alarms are triggered almost simultaneously. It is likely that the alarm sequences belonging to Cluster 0 are not a consequence of a disturbance propagating through the plant since there is no significant time delay between most of the alarms. In fact, the condition of the SUO alarm (Substitute value active) that occur when a port of an online analyzer is malfunctioning. If an online analyzer is deficient each of its ports will report an alarm separately. Although this kind of alarm floods sequences cannot be further analyzed by the proposed method, these findings are useful for the end user as they might suggest a change on the approach of alarm settings for this kind of devices, e.g. if the device fails it would be better to generate a single alarm notifying it rather than generating an alarm for each one of its ports.

Two alarm sequences taken from Cluster1 and Cluster0 are compared respectively in Figure 7.12 and Figure 7.13. The plots show the similarity between the pairs of alarm sequences and the existence of time delays between alarm occurrences. It is then likely that these alarm sequences are a consequence of a disturbance propagating through the plant. One can additionally notice that a large portion of the alarms in these sequences are limit alarms (*Condition: AHY01*, *AHHY01*, *ALY01*, *ALL01*).

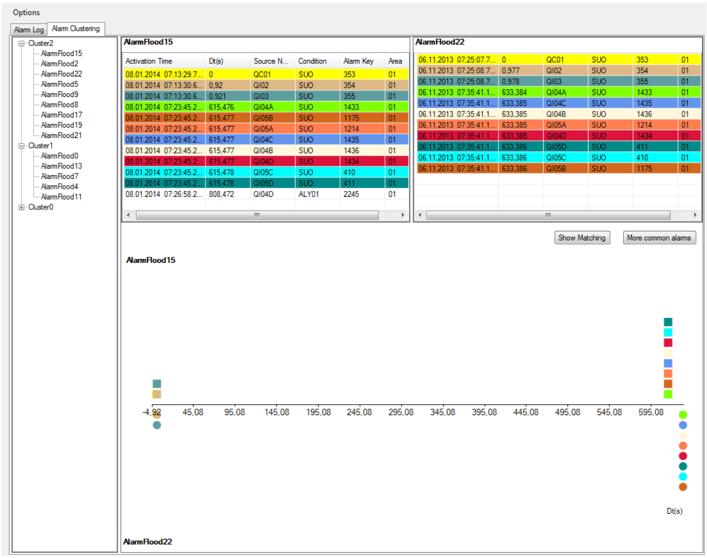


Figure 7.11 Alarm flood sequences taken from Cluster 2 are visually compared

Finally, the obtained alarm templates of the three fault scenarios corresponding to Cluster 0 and Cluster 1 are displayed in Figure 7.14.

In summary, a total of 23 alarm flood sequences have been found in the alarm log of the area 01. Out of these 23 alarm sequences, 19 are grouped under 3 clusters the other four alarm flood sequences are not similar to the rest. The alarm sequences under Cluster 2 are not caused by a disturbance propagating through the plant, therefore these alarm sequences are not further analyzed in this work.

7.3.3 Root-cause analysis

In this section the root-cause analysis is described. First process data analysis is performed and a root-cause diagnosis is suggested. The ABB PDA routines implemented in the process data analyzer of the developed tool are used for this purpose.

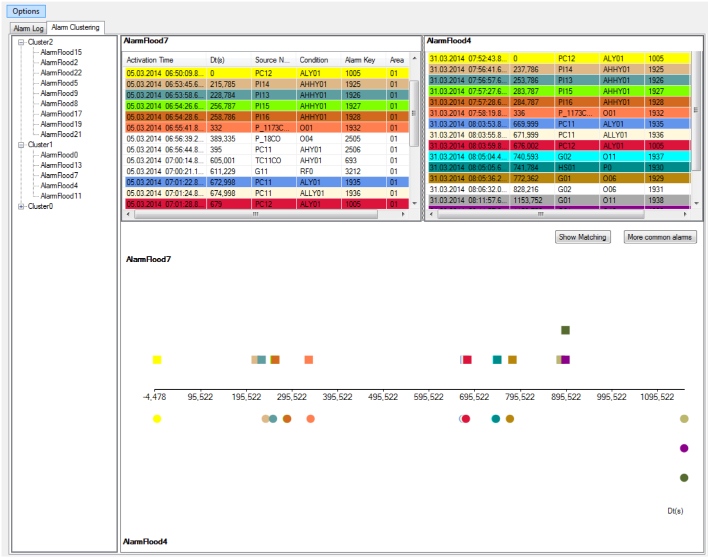


Figure 7.12 Alarm flood sequences taken from Cluster 1 are visually compared

Later, the root-cause suggestion is validated using the process topology with the Topology analyzer functionality included in the implemented software. Assuming that alarm flood sequences belonging to the same cluster have the same root-cause, the root-cause analysis just is done for a single alarm flood sequence taken from each cluster. The selected alarm flood sequence is AlarmFlood0 belonging to Cluster1.

Process Data analysis A data set with one-second sampling period is used in this analysis. The process measurements included in the process data file belong to the selected plant area The file dataset consists of time-stamps in the first column and the values of the signal measurements in the following columns.

In order to be able to perform the data-driven root-cause analysis, the signals associated to the alarms included in the template are identified.

“Process data analyzer” can be accessed by clicking on the tab “Signal Analysis”. After loading the process dataset from an Excel file, the template corresponding to Cluster1 is selected and by pressing the button “Map alarms to signals”, the automatic process alarm/process signal mapping result is obtained as shown on the mapping TreeView in Figure 7.15.

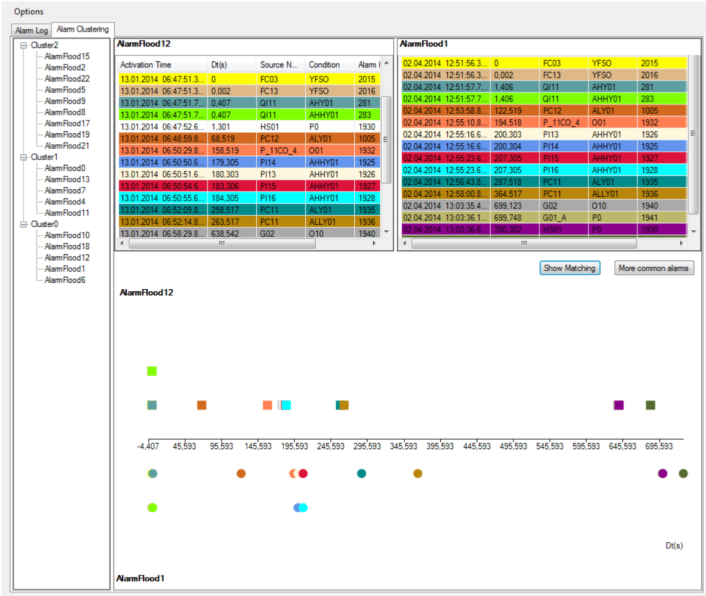


Figure 7.13 Alarm flood sequences taken from Cluster 0 are visually compared

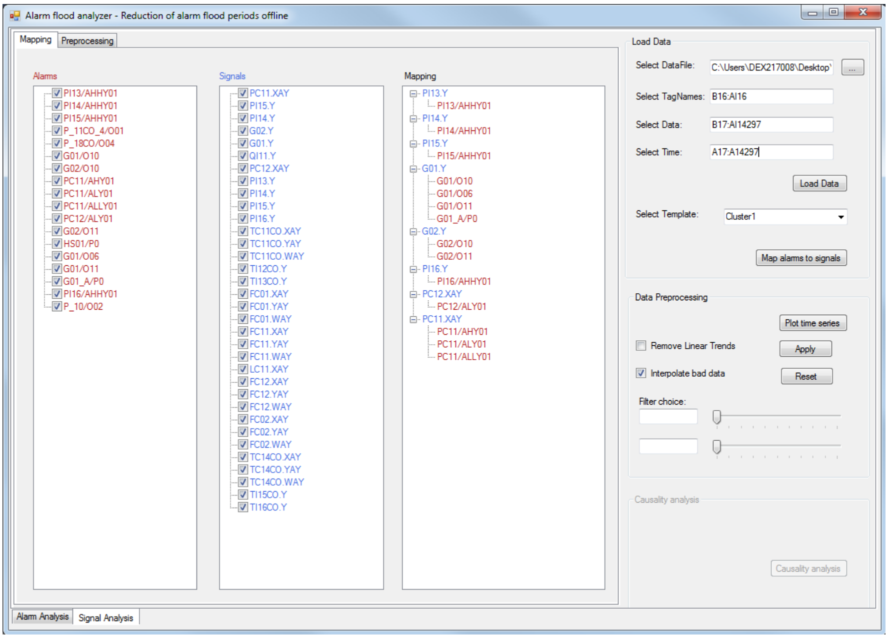


Figure 7.15 Automatic mapping

Fault Template:

Template Cluster0

Source N...	Condition	Alarm Key	Area
FC03	YFSO	2015	01
FC13	YFSO	2016	01
G01_A	P0	1941	01
G02	O10	1940	01
G02	O11	1937	01
HS01	P0	1930	01
P_10	O02	967	01
P_11CO_4	O01	1932	01
PC11	ALLY01	1936	01
PC11	ALY01	1935	01
PC12	ALY01	1005	01
PI13	AHHY01	1926	01
PI14	AHHY01	1925	01
PI15	AHHY01	1927	01
PI16	AHHY01	1928	01
QI11	AHHY01	283	01
QI11	AHY01	281	01
TC11CO	AHY01	693	01

Fault Template:

Template Cluster1

Source N...	Condition	Alarm Key	Area
G01	O11	1938	01
G01	O06	1929	01
G01	O10	1939	01
G01_A	P0	1941	01
G02	O11	1937	01
G02	O10	1940	01
HS01	P0	1930	01
P_10	O02	967	01
P_11CO_4	O01	1932	01
P_18CO	O04	2505	01
PC11	ALLY01	1936	01
PC11	ALY01	1935	01
PC11	AHY01	2506	01
PC12	ALY01	1005	01
PI13	AHHY01	1926	01
PI14	AHHY01	1925	01
PI15	AHHY01	1927	01
PI16	AHHY01	1928	01

Figure 7.14 Fault templates. (a) Template Cluster0 (b) Template Cluster1

By visual inspection, one can notice that the tag mapping algorithm gives good results. Only the alarm with tag “P_11CO_4/O01” is not mapped to its corresponding signal (PC11.XAY). In order to add this alarm to the alarms mapped to PC11.XAY signal, the alarm node is manually dragged under the signal node in the mapping TreeView.

The process signals appearing in the mapping TreeView are considered for the signal analysis (see 7.16). After pressing the button “Causality analysis”, the results of the causality analysis are displayed on the “Process data analysis result” (see 7.16).

Those signal tags within the mapping TreeView will be considered in the signal analysis, Figure 7.16. After pressing the button “Causality analysis”, the results of the causality analysis are displayed on the “Process data analysis result” tab (top), see Figure 7.17.

The bubble chart suggests PC11.XAY as the close to the root-cause. GI9122.Y, PC12.XAY, PI15.Y, PI14.Y, PI13.Y, PI16.Y and GI9123.Y as secondary disturbed signals.

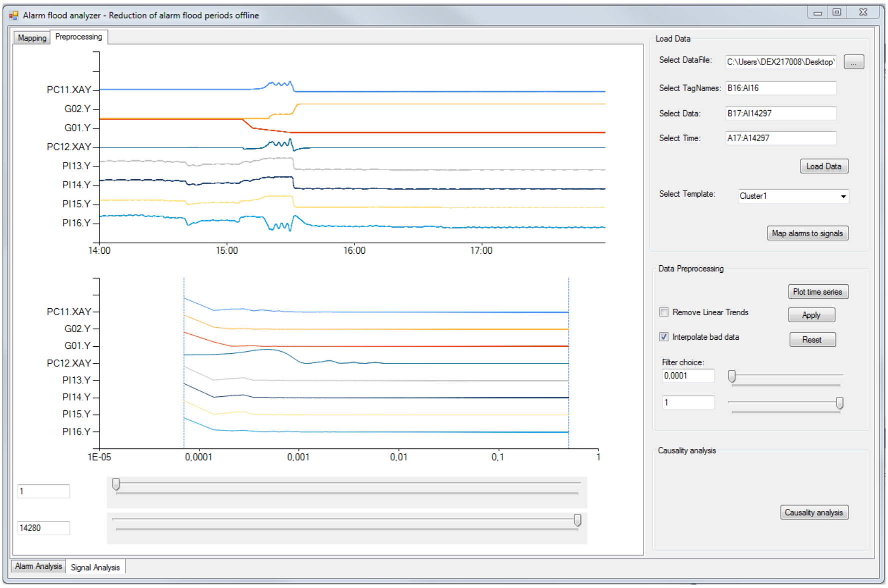


Figure 7.16 Time trends and spectra of the tags included in the analysis

Topology analysis The topology analysis is used to validate the results of the signal analysis. The process plant connectivity analysis evaluates if there is a feasible propagating path from the hypothesized root-cause to all other secondary affected assets.

The topology analyzer is accessed by clicking on the tab “Topology Analysis” (Figure 7.18-bottom side).

The assets associated to the signals used in the process data analysis and the assets associated to the alarms within the fault template are first identified. Once the process topology information is loaded in the tool, the assets TreeView is automatically populated with the asset tags found in the loaded topology model. By pressing the button “Map to assets” the automatic mapping between signals and assets, and alarms and assets is performed (See Figure 7.18 below).

By pressing the button “Load Mapping”, the suggested root-cause asset and the suggested secondary disturbed assets are automatically loaded into the “Check connections to the RC” fields. The “Path analysis” option will not be used in this case study since it is not contemplated in the method.

The validation is carried out by pressing on the “Check Connections with RC”

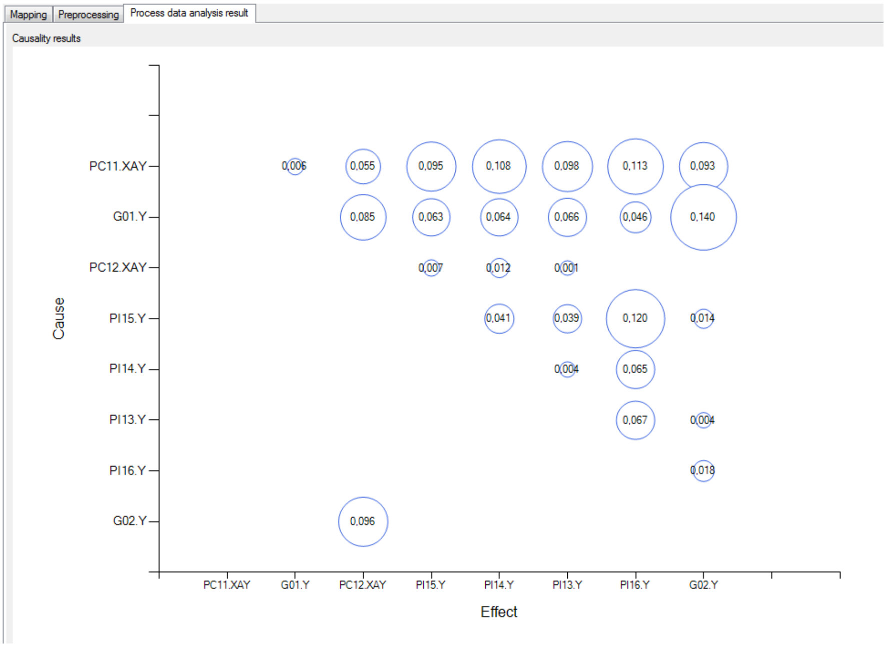


Figure 7.17 Causality matrix

button. Results are displayed in Figure 7.19 and 7.20.

The plant topology analysis indicates that there exists a feasible propagation path starting from the suggested root cause to all secondary disturbed points. In addition, the graphical visualization depicts the obtained feasible propagation paths. The graph suggests that the fault propagated from the pressure controller PC11 to the valves G01 and G02 via the information connection between this controller and the valves. From valve G01, the disturbance spread to the pressure controller PC12. And in parallel through the four cracked gas lines to the furnaces where Cracking Zone A, Cracking Zone B, Cracking Zone C and Cracking Zone D are located (via the heat exchangers LQE 1, LQE 2, LQE 3, LQE 4). Sensors PI13, PI14, PI15 and PI16 are also affected by the propagating disturbance they are located in the coils of the different cracking zones.

The root-cause suggestion from the data-driven analysis is therefore validated by the plant topology. The connectivity analysis has validated that the pressure controller PC11 is a valid root-cause of the obtained situation corresponding to the alarm flood sequences in Cluster1. The alarm associated to this controller is the given causal alarm suggestion, i.e., alarm PC11_CO_4. All the alarms in the

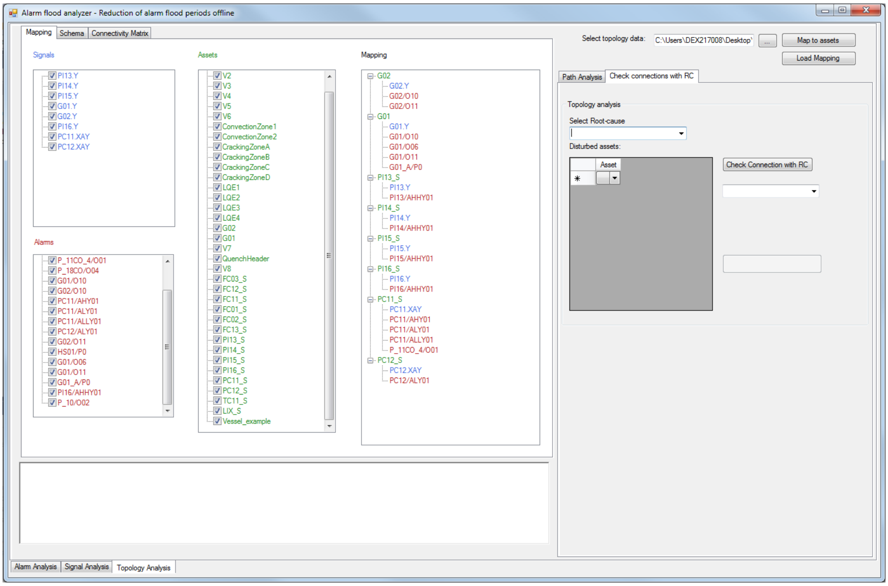


Figure 7.18 Mapping to assets

template will be grouped under it.

7.3.4 Discussion of the results

The use of a combination of multiple information sources has proved to be beneficial in order to not only identify alarms caused by the same fault, but also to isolate the causal alarm within an alarm flood sequence. The evaluation of the alarm log showed that a specific area within the ethylene plant experienced a significant number of alarms floods caused by a disturbance propagation through the plant. The alarm analysis presented in this section reveals the existence of two common abnormal situations that lead to alarm flooding. One of these abnormal situations was further analyzed using process data and plant connectivity data. A root-cause of the fault was suggested by the data-driven analysis and validated by plant topology. The causal alarm of the alarm flood sequence that characterizes this fault was identified, PC11_CO_4.

The obtained results were validated by the site process control engineers. The two abnormal situations identified in plant area 01 correspond to a mode operation shift between cracking-decoking and decoking-cracking. The transition between these two modes of operation triggers characteristic alarm flood sequences. The alarm flood sequences grouped under Cluster1 correspond to the cracking-decoking

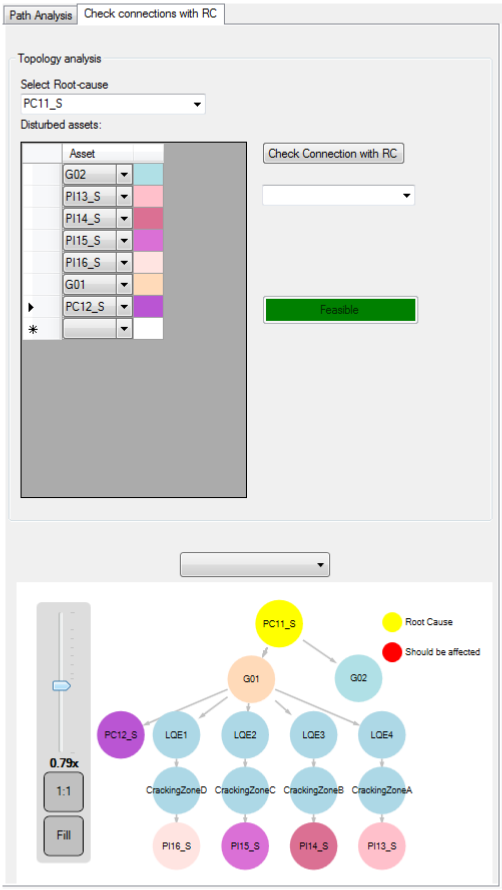


Figure 7.19 Graphical visualization of the obtained feasible propagation paths

transition while those belonging to Cluster 0 are associated to the decoking-cracking transition.

The site crew mentioned that pressure controller PC11 is the controller performing the opening and closing of both valve G01 and valve G02 during the change of operation mode. Its goal is to prevent any backflow of the cracked gas into the furnace. The transition from cracking to decoking operation mode, the one analyzed, follows the following sequence:

- Activate control loop PC11-G01 with a fixed differential pressure set point.
- While the valve G02 stays closed, the valve G01 is driven towards its closed position.

- When a specified differential pressure is reached, the control loop PC11-G02 is activated with the same differential pressure set point.
- The controller drives the valve towards its open position .
- Once valve G01 is closed, the control loop PC11-G02 is deactivated and the valve G02 is driven at constant speed until it is completely closed.

This information confirmed our findings: First the pressure controller PC11 is activated and a set point is entered, this triggers alarms associated to the controller. The controller drives the valve G01 which triggers alarms connected to it. Pressure is built in the cracking areas due to the closing of the valve and therefore the pressure alarms connected to the measurement points PI13, PI14, PI15 and PI16 occur. Once a given pressure is reached, the controller starts moving valve G02 triggering alarms associated to it. Pressure controller PC12 manipulates the steam flow that assures the sealing of valve G01 once it is closed. This controller maintains the differential pressure in the valve at a specified value. When the differential pressure drops an alarm is triggered.

Even though the alarm floods associated to the operational mode shift are the result of an operator action. The process experts wanted an alarm reduction for this operation. In the event of an incident during the transition, the operators will then be able to focus on the incident and will not be distracted by the alarms related to the operation mode change. Furthermore, according to the plant operators, the ethylene plant works smoothly and upsets are not something common. The plant usually has just one or two abnormal situation in half a year. In order to isolate alarm floods associated to the critical scenarios, a longer alarm log should be analyzed. The proposed method was able to identify and reduce successfully the alarm floods associated to the cracking-decoking shift without previous awareness.

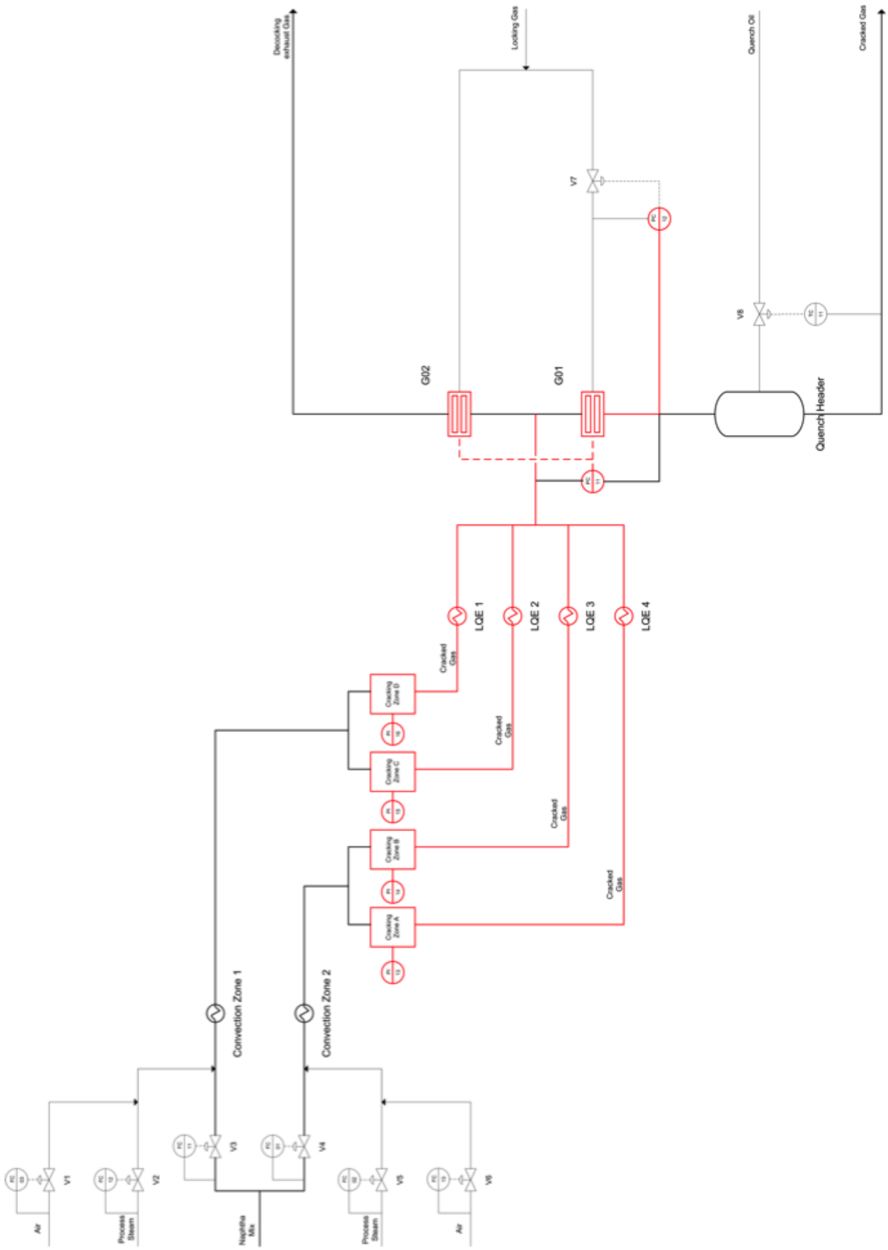


Figure 7.20 Schema with the path highlighted

8

Conclusions

8.1 Contributions

Alarm floods represent an obstacle for the correct supervision of process plants. They are a source of plant incidents and have become a serious problem in the process industry. This thesis presents an innovative way to reduce alarm flooding by using a combination of information sources. Alarm log analysis, process data analysis and connectivity analysis are used to not only group consequential alarms originating from the same process abnormality, but also give a causal alarm suggestion.

The work reported in this thesis has been engineered into a software tool that guides the user through the different steps of the method. A case study with real industrial data has been presented to demonstrate the utility of the method and the software tool developed.

8.2 Future Work

In this work, a method that reduces alarm floods off-line has been presented. A reconfiguration of the alarms involved in each of the process abnormalities analysed could prevent future alarm floods related to these abnormalities. Another direction of the future work will be to study the applicability of the process for on-line alarm flood reduction. The authors of the method are aware of the high computational burden of the presented solution. This makes the direct applicability on-line infeasible. However, the fault templates obtained off-line could be use to identify the process abnormality causing an incoming alarm flood. Similarities between the current alarm flood and the fault templates can be extracted. When the incoming alarm flood presents a high similarity with one of the fault templates, the causal alarm of this template could be assigned to the new alarm flood and all the incoming alarms present in the fault could be grouped under the causal alarm.

Bibliography

Ackermann, M., J. Blömer, D. Kuntze, and C. Sohler (2012). "Analysis of agglomerative clustering." *Algorithmica*.

AGDAS, D. and R. D. ELLIS (2010). "The potential of xml technology as an answer to the data interchange problems of the construction industry." *Construction Management & Economics*.

Ahmed, K (2011). *Similarity analysis of industrial alarm flood data*.

Ahnlund, J., T. Bergquist, J. E. Larsson, and L. Spaanenburg (2003). "Intelligent reduction of nuisance alarms in process control".

ANSI/ISA-18.2 (2009). "Management of alarm systems for the process industries".

Barth, M., M. Strube, A. Fay, P. Weber, and J. Greifeneder (2009). "Object-oriented engineering data exchange as a base for automatic generation of simulation models." In:

Bauer, M., N. Thornhill, J. Cox, M. Caveness, and J. Downs (2007). "Finding the direction of disturbance propagation in a chemical process using transfer entropy." *IEEE Transactions on Control Systems Technology*.

Bergquist, T., J. Ahnlund, and J. Larsson (2003). "Alarm reduction in industrial process control". *Proceedings of the EFTA 2003. 2003 IEEE Conference on Emerging Technologies and Factory Automation*.

Bouchair, N., P. Gayet, and S. Charbonnier (2013). "Fault isolation by comparison of event lists using a weighted distance". *IEEE Internation Conference on Systems, Man and Cybernetics*.

Bransby, M. L. and J. Jenkinson (1998). "The management of alarm systems: a review of best practice in the procurement, design and management of alarm systems in the chemical and power industries". *HSE Research Report CRR 166*.

Cai, R., Z. Hao, Z. Zhang, A. Tung, and C. Dai (2014). "A general framework of hierarchical clustering and its applications." *Information Sciences*.

Cheng, Y., T. Chen, and I. Izadi (2013). “Pattern matching of alarm flood sequences by a modified smith-waterman algorithm.” *Chemical Engineering Research and Design*.

Christofides, P., J. Davis, N. El-Farra, D. Clark, K. Harris, and J. Gipson (2007). “Smart plant operations: vision, progress and challenges.” *AIChE Journal*.

Duisting Beebe, S. F. and D. Logerot (2007). “Alarm floods and plant incidents”. *ProSys*.

EEMUA-191 (2007). *Alarm systems: a guide to design, management and procurement, Version 2*.

Fan, Y. and X. Deyun (2012). “Progress in root cause and fault propagation analysis of large-scale industrial processes.” *Journal of Control Science & Engineering*.

Fan, Y., X. Deyun, and S. L. Shah (2010). “Qualitative fault detection and hazard analysis based on signed directed graphs for large-scale complex systems”.

Fang Yang, D. X. (2012). “Progress in root cause and fault propagation analysis of large-scale industrial processes”. *Journal of Control Science and Engineering*.

Gao, D., B. Zhang, X. Ma, and C. Wu (2010). “Signed directed graph and qualitative trend analysis based fault diagnosis in chemical industry.” *Chinese Journal of Chemical Engineering*.

Henningsen, A. and J. P. Kemmerer (1995). “Intelligent alarm handling in cement plants”. *IEEE Industry Applications Magazine*.

Higuchi, F., I. Yamamoto, T. Takai, M. Noda, and H. Nishitani (2009). *Use of event correlation analysis to reduce number of alarms.* Computer Aided Chemical Engineering.

Hollender, M. (2010). *Collaborative Process Automation Systems*. ISA.

Hollifield, B. R. *Alarm Management: A Comprehensive Guide, Second Edition*. Isa.

Hugo, A. (2009). “Estimation of alarm deadbands”. *Proceedings of the 7th IFAC SAFEPROCESS, Barcelona, Spain*.

IEC (2008). “Representation of process control engineering - requests in p&id-diagrams and data exchange between p&id tools and pce-cae tools”. *IEC 62424*.

Kabir, A., I. Iyadi, T. Chen, D. Joe, and B. T. (2013). “Similarity analysis of industrial alarm flood data.” *IEEE Transactions on automation science and engineering*.

Li, H. (2000). “Xml and industrial standards for electronic commerce.” *Knowledge and Information Systems*.

Naghoosi, E., I. Izadi, and C. Tongwen (2011). "A study on the relation between alarm deadbands and optimal alarm limits". *American Control Conference (ACC), 2011*.

Nishiguchi, J. and T. Takai (2010). "Ipl2 and 3 performance improvement method for process safety using event correlation analysis." *Computers and Chemical Engineering*.

Rothenberg, D. (2009). *Alarm management for process control*.

Schleburg, M., L. Christiansen, A. Fay, and N. Thornhill (2013). "A combined analysis of plant connectivity and alarm logs to reduce the number of alerts in an automation system." *Journal of Process Control*.

Schleipen, M., O. Sauer, and R. Drath (2008). "The system-independent data exchange format caex for supporting an automatic configuration of a production monitoring and control system." In: *IEEE International Symposium on Industrial Electronics*.

Schreiber, T. (2000). *Measuring information transfer*. Phys. Rev. Lett.

Seema, B. and E. Coyle (2003). "An energy efficient hierarchical clustering algorithm for wireless sensor networks." *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer & Communications Societies (IEEE Cat. No.03CH37428)*.

Smith, T. and M. Waterman (1981). "Identification of common molecular subsequences." *Journal of Molecular Biology*.

Thambirajah, J., N. Thornhill, L. Benabbas, and M. Bauer (2009). "Cause-and-effect analysis in chemical processes utilizing xml, plant connectivity and quantitative process history." *Computers and Chemical Engineering*.

Thornhill, N. (2005). "Finding the source of nonlinearity in a process with plant-wide oscillation." *IEEE Transactions on Control Systems Technology*.

Thornhill, N., S. Shah, B. Huang, and A. Vishnubhotla (2002). "Spectral principal component analysis of dynamic process data". *Control Engineering Practice*.

Thornhill, N., B Huang, and H Zhang (2003a). "Detection of multiple oscillations in control loops". *Journal of Process Control*.

Thornhill, N. F., J. W. Cox, and M. A. Paulonis (2003b). "Diagnosis of plant-wide oscillation through data-driven analysis and process understanding". *Control Engineering Practice*. Award winning applications-2002 {IFAC} World Congress.

Venkatasubramanian, V., R. Rengaswamy, and S. Kavuri (2003). "A review of process fault detection and diagnosis. part ii: qualitative models and search strategies." *Computers and Chemical Engineering*.

Wang, J. and T. Chen (2014). "An online method to remove chattering and repeating alarms based on alarm durations and intervals." *Computers & Chemical Engineering*.

Williams, W. T. (1975). "Numerical taxonomy. the principles and practice of numerical classification. peter h. a. sneath robert r. sokal." *The Quarterly Review of Biology*.

Yang, F., S. Shah, D. Xiao, and T. Chen (2012). "Improved correlation analysis and visualization of industrial alarm data." *ISA Transactions*.

Yuan, T. and S. J. Qin (2014). "Root cause diagnosis of plant-wide oscillations using granger causality." *Journal of Process Control*.

Zhengzheng, X., J. Pei, and E. Keogh (2010). "A brief survey on sequence classification." *ACM SIGKDD Explorations Newsletter*.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER´S THESIS

Date of issue
September 2014

Document Number
ISRN LUTFD2/TFRT--5961--SE

Author(s)
Vicent Rodrigo Marco

Supervisor
Moncef Chioua, ABB
Tore Hägglund, Dept. of Automatic Control, Lund University, Sweden (examiner)

Sponsoring organization

Title and subtitle
Alarm flood reduction using multiple data sources

Abstract

The introduction of distributed control systems in the process industry has increased the number of alarms per operator exponentially. Modern plants present a high level of interconnectivity due to steam recirculation, heat integration and the complex control systems installed in the plant. When there is a disturbance in the plant it spreads through its material, energy and information connections affecting the process variables on the path. The alarms associated to these process variables are triggered. The alarm messages may overload the operator in the control room, who will not be able to properly investigate each one of these alarms. This undesired situation is called an “alarm flood”. In such situations the operator might not be able to keep the plant within safe operation. The aim of this thesis is to reduce alarm flood periods in process plants. Consequential alarms coming from the same process abnormality are isolated and a causal alarm suggestion is given. The causal alarm in an alarm flood is the alarm associated to the asset originating the disturbance that caused the flood. Multiple information sources are used: an alarm log containing all past alarms messages, process data and a topology model of the plant. The alarm flood reduction is achieved with a combination of alarm log analysis, process data root-cause analysis and connectivity analysis. The research findings are implemented in a software tool that guides the user through the different steps of the method. Finally the applicability of the method is proved with an industrial case study.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-96

Recipient’s notes

Security classification