University of Alberta

Similarity Analysis of Industrial Alarm Flood Data

by

Kabir Ahmed

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Controls

Department of Electrical and Computer Engineering

Fall 2011
Edmonton, Alberta

To my parents

Abstract

Alarm floods are a crucial problem in the process industry. An alarm flood makes it difficult for an operator to react and take necessary actions, which often can lead to risking an emergency shutdown or a major upset. In many cases, alarm floods are caused by interrelated process variables, which can be identified via similar patterns in alarm annunciations. This similarity can be investigated through alarm pattern analysis of industrial alarm flood data.

In this work, alarm floods are discussed based on the standards presented in the new ISA 18.2 guidelines and the discussion given in EEMUA 191. A new analysis method is proposed to identify alarm floods that are similar from the historical alarm data and group them on the basis of patterns of alarm occurrences. Patterns in alarm sequences can be investigated through different distance measures. To calculate a distance between alarm patterns in two different sequences, preprocessing of industrial alarm data and effective flood period isolation are required. Hence, definitions of alarm floods and alarm flood periods are given based on the new ISA 18.2 standards. The effect of chattering alarms on alarm floods is also discussed. Three different distance scores, suitable for capturing alarm patterns in alarm flood sequences, are introduced.

Finally, a case study on real industrial alarm data is presented to demonstrate the utility of the proposed analysis.

Acknowledgements

I seek the opportunity to extend my sincere thanks to all the individuals without whom this thesis would not have been possible.

- First of all, I especially would like to express my sincere gratitude to my supervisor Dr. Tongwen Chen (Professor, Electrical and Computer Engineering Department, University of Alberta), for his guidance during my research and study at the University of Alberta. His perpetual energy and enthusiasm in research had made the momentum of this research going. I would like to thank him for being always available and willing to help his students with their research in his busy schedule.

- A deep thankful gratitude to Dr. Iman Izadi (PDF and Research Manager, Advances in Alarm Management and Design research group) for his valuable suggestions, support, and motivation on the research and help in conducting the industrial case study.

- I would like to thank Dr. Sirish Shah (Professor, Chemical and Materials Engineering Department, University of Alberta), and all the members of Advances in Alarm Management and Design research group for their insightful comments and encouragement.

- I acknowledge and thank the National Sciences and Engineering Research Council of Canada, Strategic Project Grant program, for their financial support in this work. I also wish to thank Imperial Oil, Suncor Energy, and Matrikon Inc (now part of Honeywell Advanced Solutions) for their collaboration in this research.

- My earnest thanks to my thesis examination committee members, Dr. Mahdi Tavakoli, and Dr. Sirish Shah for their careful reading of this thesis and perceptive suggestions.

- Last but not the least, my heartfelt gratefulness to my family and my colleagues. My special thanks to Salman Ahmed, Ghulam Mustafa, Naseeb Adnan, Geoff McDonald, and Md. Ruhul Kabir for their help in preparing this thesis.

My sincere apology for omissions of any contribution, which may have occurred unintentionally.

Kabir Ahmed

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Process Industry and Alarm Systems

The economic prosperity of the process industry depends on the profit earned by its core business operations. To maximize this prosperity, optimal plant operation is very important. Most of the research and development on optimal plant operations, which have been carried out for the last few decades, led towards monitoring and regulating more and more process variables. As a result, modern industrial plants contain a large number of sensors and actuators communicating with hundreds of clients and control loops. In a smart plant operation, the operating profit is expected to be maximized ensuring a sustainable environmental, health and safety (EH&S) performance [2]. In this paradigm, implementation of a self evaluated and non-hierarchical operation with respect to information flow was suggested (Figure 1.1). But unfortunately, it comes at the cost of supporting a high number of interactions among process variables.

Since the introduction of Distributed Control Systems (DCS) and with the advancement in industrial computer and communication technology, it has become very easy to configure thousands of process variables and support high interactions among them. As a result, it is often found that in a plant, a single fault in one component can produce inconsistent outputs which serve as input excitations to many other healthy parts of the plant. This type of interconnections are very likely to result in cascaded faults in a system risking the plant to upsets. According to the Abnormal Situation Management (ASM) Consortium [3], petrochemical plants on average suffer a major accident once every three years. A good number of hazardous incidents in the process industry have been reported, resulting in plant

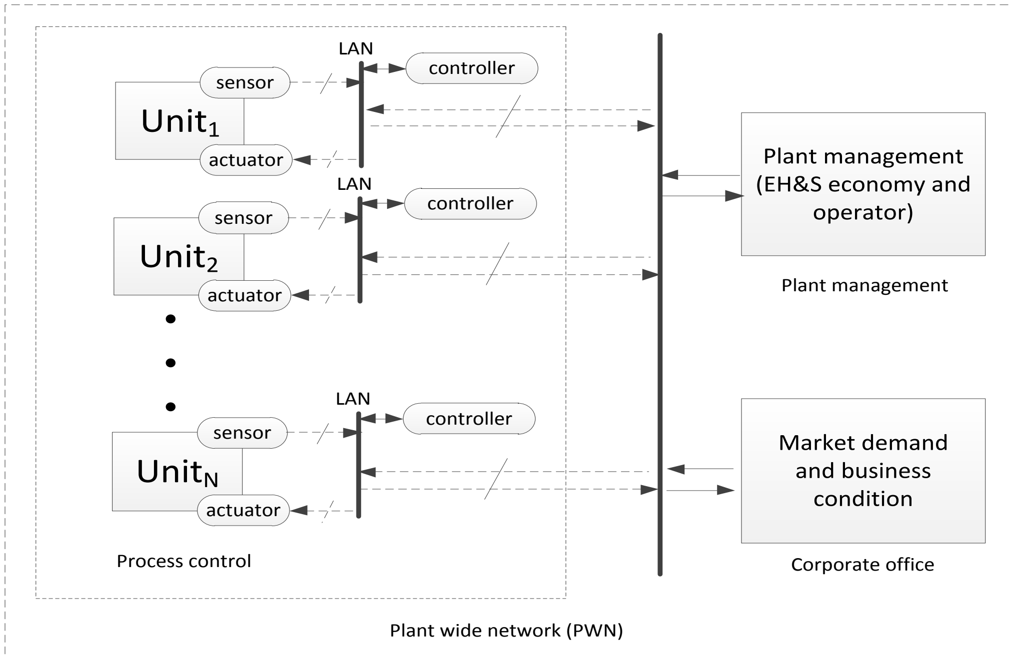


Figure 1.1: Smart plant operation paradigm

damages, loss of production, and environmental degradations due to poor performance of alarm systems.

Typically, an alarm is raised if any process variable crosses the corresponding alarm limit. An alarm limit is the safety boundary of the operation range determined on the basis of limitation of equipments, product quality, and safety issues concerning plant's assets. The core purpose of an alarm system is to alert operators if a process variable violates any of the associated safety boundaries. It serves as the communication link between a process and an operator. Without an alarm system, a process is the same as without any protection at all. A quantitative illustration of the contribution of an alarm system was presented in [4] showing that an alarm system is four times more significant than a trip system to protect a plant from hazards.

Owing to software alarms introduced by modern DCS, alarms are now very easy to configure with absolutely no extra cost and very little changes in DCS settings. This facilitates the setting of alarms at each and every possible point in a plant. Often, many alarms are configured without proper analysis and rationalization. Consequently, plant operators frequently experience high alarm counts with considerably large portions of false and nuisance alarms [5, 1]. In some cases, the volume of alarms presented to an operator is so high that it exceeds his/her response capability. Such events are commonly known as *alarm floods* or *alarm showers* [6].

Ideally, every alarm should precisely draw the attention of an operator and convey only related information where the operator's attention is required. Whereas, during an alarm flood it is exactly opposite to what is expected in the ideal case. An alarm flood causes operators to be overwhelmed by a large volume of alarms. As a result in most of the cases operators fail to respond accordingly. Although alarm floods have been on the top priorities from the industrial point of view, not much research has addressed this particular problem. The quantitative definition of an alarm flood also varies in different industries in practice. Hence, a research work focused on establishing a benchmark on analysis of alarm floods is very much required. This is the main focus of this thesis.

1.2 Alarm Management and Alarm Flood Analysis

1.2.1 Standards in alarm management

The rapid advances in technology and control systems have resulted in the process industry having to adapt to significant changes in plant operations worldwide. Hence, to establish uniformity in definitions, practice, and performance, many organizations were formed. Two of such organizations which have been widely accepted by most of the industries across Asia, Europe and America are the *International Society of Automation (ISA)* and the *Engineering Equipment and Materials Users Association (EEMUA)*.

The *International Society of Automation (ISA)* has provided standards in automation for over 65 years. Recently, a significant milestone in alarm management was announced by the ISA through the publication of ISA 18.2 Standards, "Management of Alarm Systems for the Process Industries", which has been approved by the ISA Standards & Practices Board and American National Standards Institute (ANSI). The primary purpose of ISA 18.2 is to provide standards in practice of alarm systems including definitions, design, management, installation, and effective processing to meet high quality in performance in an alarm management lifecycle.

A similar guideline on alarm management was published by EEMUA entitled "Alarm Systems: A Guide to Design, Management and Procurement (EEMUA 191)" [7]. EEMUA 191 provides guidelines to review, design, and prioritize alarms in industry. It also provides benchmarks on performance of alarm systems.

Both ISA 18.2 and EEMUA 191 have discussed various efforts required for effective design and maintenance of alarm systems. This includes stages in alarm management life cycle, design principles, implementation issues, performances of alarm systems, maintenance, and improvements. In recent research, these discussions have been found to be very useful. Hence, it is important to present a brief discussion on alarm floods in the light of ISA 18.2 and EEMUA 191 guidelines.

1.2.2 Alarm flood analysis in ISA 18.2 and EEMUA 191

According to EEMUA 191, under normal conditions an operator needs approximately 10 minutes to manage an alarm effectively and the maximum rate of alarms per hour should not exceed 60 alarms per hour. An alarm rate above this limit is difficult to manage and most likely to be perceived as a flood of information by a human operator.

In the process industry, alarm system performance metrics, which reflect various measures of alarm information rates to operators, are significantly higher than the standards. A comparison between EEMUA suggestions and the average values of the performance metrics currently prevalent in oil and gas, petrochemical, and power industries are shown in [1] as reproduced in Table 1.1:

Table 1.1: EEMUA benchmark and average values received in industries [1]

	EEMUA	Oil and Gas	Petrochemical	Power
average alarms/hour	≤ 6	36	54	48
average standing alarms	9	50	100	65
peak alarms/hour	60	1320	1080	2100
distribution % (low/med/high)	80/15/5	25/40/35	25/40/35	25/40/35

In ISA 18.2, it is also recommended that an alarm system should not be in flood for greater than one percent of the total reporting time. However, in reality it is much higher than this requirement. Making only improvement in individual alarm design might not be enough to resolve this problem. Hence, in ISA 18.2, it is suggested to consider advanced and enhanced techniques for alarm flood study and analysis.

Depending on the degree of complexity involved, advanced and enhanced alarming techniques can be categorized into different classes as follows:

- Information linking, (e.g., similarity analysis of alarms)
- Logic based alarming (e.g., logical suppression, state based alarms),
- Model based alarming, (e.g., techniques which uses both process data and process knowledge) and
- Additional alarming consideration (e.g., utilizing auxiliary and remote alarm systems)

A routine analysis of alarm floods is also very important. A periodic check on process operation can reveal the state of improvements or degradation in alarming. Different performance metrics suggested by ISA 18.2 to capture improvements or degradations in alarm systems, are as follows:

- Number of alarm floods per reporting period,
- Duration of each alarm flood,
- Alarm count of each alarm flood, and
- Peak alarm rate for each alarm flood.

These quantitative measures of alarm flood characteristics are well adopted by the petrochemical industry as a part of their routine alarm flood analysis. But as mentioned before, these performance metrics capture only the state of an alarm system. Information which can be used to compare different floods in terms of process dynamics and mutual interaction among the process variables are not reflected on these performance metrics.

In many cases, alarm floods are caused as a result of poorly rationalized alarms and inappropriate alarm design. As a result, it has been observed that a large portion of alarms raised in an alarm flood are inter-related and follow a very specific pattern in their sequence of occurrences. Often a group of alarms is raised as a result of another specific set of alarms and maintain a specific sequence in their annunciations. To capture these similarities, classification of recorded alarm floods is a key step to consider for a systematic analysis.

A similarity investigation on alarm floods is an advanced alarming technique which falls under the category of *'Information Linking'*. Here, the information from master alarm database is used to find patterns in alarm annunciations for each recorded flood. Depending on the similarities in the patterns of alarm occurrences, a similarity or a distance measure can be computed to group different alarm floods recorded over time. This analysis will facilitate the finding of the root cause in an alarm set where alarms are interrelated and raised in sequences. If the root cause of a specific alarm set is known in advance, operators can be given priorities on alarms and they would be able to take necessary actions during a similar alarm flood in the future.

1.3 Literature on Alarm Management

Alarm management and design form a relatively new area in research. Until recently, not much research has specifically focussed on alarm systems in the process industry. Some of the early work related to alarm processing has been published in various sectors other than the chemical processes and control engineering, such as nuclear science [8], computer intrusion detection [9], and power systems [10, 11]. Recent research in alarm management and design for the process industry can be divided into two categories: single variable alarm design, and multivariate alarm processing and analysis.

A single variable alarm design is a primary but necessary step to an optimal and effective alarm system. In general, variables in industries are two types: process variables (PVs) or manipulated variables (MVs) [12]. Typically, these variables on which alarms are set have wide dispersions in their statistical distributions. Commonly identified reasons for such a behavior are: presence of noise, different operating conditions, and instrumentation and measurement limitations. As a result, a straightforward setting of alarm limits may cause too many alarms with a considerably high false and missed alarm rates [5, 1, 8]. To overcome this problem, many signal processing techniques have been suggested and studied in many different publications [5, 13, 14]:

- Filtering of process data,
- Delay timer in raising and clearing alarms,
- Deadbands,
- Alarm window design.

In [15], alarm limit optimality in the sense of a *Receiver Operating Characteristic (ROC)* is studied. A *Receiver Operating Characteristic* is a plot of the missed alarm rate against the false alarm rate and is very effective in analyzing performance of an alarm design. It captures the tradeoff between the false and missed alarm rates and helps to design alarms optimally with minimum false and missed alarm rates. In the aforementioned study, different forms of optimal filters (linear and nonlinear) for reduced false and missed alarm rates are discussed. Different distributions of variables are also considered in the design of the optimal filters.

Delay timer and *deadbands* are two of the most effective techniques in reducing missed and false alarm rates. But, like every other signal processing technique, it comes at the cost of a significant *detection delay* and if not considered properly may lead to severe losses of plants assets. In [16], computation of *detection delays* for alarms with delay timers and dead bands is discussed. There exists salient relationship between *deadbands* and alarm limits. Hence it should be carefully considered while using *deadbands* in alarms. A comprehensive discussion on such relations between alarm *deadbands* and optimal alarm limits is presented in [17].

Chattering or *repeating alarms* are the most common form of *nuisance alarms* in industries. Chattering in an alarm can be quantified based on its *run length distribution* [18]. *Run lengths* are computed from alarm data represented as binary sequences [19, 20]. More detailed discussions on binary representation of alarm data, and chattering alarms and their effect in alarm floods are presented in Chapter 2.

Typically, the number of variables in modern industry is high. In many cases it might be difficult in analyzing each individual process variable and design alarms for this variables. This motivates one to consider multivariate analysis of alarms and process variables.

The advantages of multivariate analysis over univariate alarm analysis in industries are significant. Many hidden relationships can be extracted among different variables in a process by analyzing them together. For example, in a large process, multivariate analysis can be used to extract information from many variables in the process and relate them to a few *latent* variables [1]. *Latent* variables are virtual variables which are calculated by combining raw process variables linearly. In [21], it is shown that the false and missed alarm rates can be minimized by setting considerably fewer alarms on PCA (*Principle Component Analysis*) based Q and T^2 statistics of many related variables.

Several techniques on graphical representations of alarm data, handling of data and their effective utilizations are presented in [19] and [22]. Such representations facilitate the grouping of alarms and variables in a process based on their interrelations [19, 23, 20].

Determination of the root cause among a group of interrelated alarms is a crucial step. One of the most interesting work in this regard is the causality analysis of interrelated variables. In a causality analysis, a cause and its effect are determined from time series data, based on information theoretic approaches [24, 25]. It can be used in chemical processes to determine the direction of fault propagation using different information entropy measures, e.g., the *transfer entropy* in [26]. Several other publications exist on investigation of causality and root cause for alarms in the process industry using *reachability* [27], *sign directed graph (SDG)*[28] and fusion of process data with process connectivity [29] .

In [30], an evaluation technique for industrial alarm systems is presented using an operators' model that mimics the behavior of a human operator. Such evaluation methods are useful in detecting points where attentions might be required in an existing alarm system. During the design process of a new alarm system, a good selection of the points where alarms are necessary is a fundamental step in plant design. Such a systematic design approach is discussed in [12].

Much of the discussed work has mentioned the problem of alarm flooding in general and provided suggestions in designing an effective alarm system that would produce less alarms. However, to the best of the author's knowledge, no research has specifically focussed on alarm flood analysis. Questionnaires were collected from different industrial representatives in order to determine the specific areas of alarm management that require immediate attention. The survey amongst key industrial sectors revealed that flooding of alarms is a major issue concerning the effectiveness of alarms systems in managing hazardous abnormal situations. Hence similarity investigation of industrial alarm flood data is proposed in this thesis which can be used to mitigate alarm floods caused by interrelated variables.

Similarity investigation and pattern analysis form a well-known area in the field of artificial intelligence and machine learning. Pattern investigation of alarm sequences is a similar problem as DNA sequence analysis in bioinformatics [31, 32, 33, 34], and timestamped event sequence matching in network intrusion detection [35, 36]. More details on pattern analysis and classifications of alarm floods are presented in Chapter 3.

1.4 Scope and Organization

1.4.1 Scope

Alarm flood analysis is a new area of research. Definitions and practices on alarm floods, and associated terminologies vary among different industries. This thesis presents a complete discussion on the definition, representation, data processing and classification of alarm floods based on the patterns of alarm occurrences.

Alarm floods in a process is unwanted in general and for an optimally designed alarm system each alarm flood should be unique in terms of fault propagation. The alarm flood analysis presented in this work focuses on investigation of classes among recorded alarm floods. A class in recorded alarm floods indicates repetition of a specific fault propagation in the process or interrelations among alarms. For such a case, a root cause analysis of the interrelated alarms will facilitate the mitigation of the class of alarm floods in future.

1.4.2 Organization

The necessary concepts in pattern analysis of alarm floods are alarm flood data processing, unsupervised clustering, and pattern similarity measures. Considering this, the thesis is organized as follows.

In Chapter 2, nomenclature on alarm systems in large processes is briefly discussed along with a standard definition of alarm floods, associated processing of master database and representation of alarm data for an effective detection and isolation of an alarm flood period. Chattering or repeating alarms constitute a major percentage of the total alarm count and should be considered critically while analyzing alarm floods. The effect and the handling of chattering alarms for alarm flood analysis are presented in this chapter with examples from real industrial data.

In Chapter 3, a brief introduction on pattern analysis and classification is presented. Pattern analysis and machine learning are very active areas of research for their numerous applications in different fields of science and technology. *'Unsupervised clustering'* is a well known clustering method. It is used to discover patterns in data for which prior knowledge is not available. A general theory of similarity investigation is also introduced in this chapter along with necessary problem definitions on the similarity analysis and distance measures of alarm floods.

In Chapter 4, the calculation of a distance measure between two alarm flood sequences is introduced. Different notions of distances between two alarm floods are presented in this chapter. Distance scores can be computed based on statistical behavior of alarm events in a sequence. For time series data, another widely used pattern analysis technique is the time normalized sequence mapping. *Dynamic Time Warping* is one such technique generally used for real valued time series data. In this work, it has been modified for the case of alarm flood sequences. The optimal mapping of elements in two sequences can be found via dynamic programming. A distance score between two floods can be calculated based on the successes in mapping symbols to symbols for the two flood sequences.

After calculating pairwise distances for different alarm floods, they can be clustered into different classes using unsupervised clustering. A case study on real industrial data is presented in Chapter 5, where the proposed analysis is applied to group similar alarm floods. Sampled data presented in this chapter was collected from a real industrial unit which is suspected to suffer from floods due to interrelated process variables.

The last chapter includes the summary and the scope of future work on the proposed analysis. A general discussion on its implication and uses is also presented in this chapter from an industrial point of view.

Chapter 2

Alarm Floods

2.1 Alarms in the Process Industry

2.1.1 Alarm management lifecycle

Alarm systems communicate with human operators and inform them about abnormalities or malfunctions at different points in a process. Basic process control and safety instrument systems use different measurements of process conditions. These measurements are sent to an alarm system where alarms are generated logically based on a defined philosophy.

Basic terminology in alarm systems and work processes in managing alarm systems effectively are the necessary concepts to address the problem of alarm flooding. In ISA 18.2 Standards, work processes in alarm management are expressed as a lifecycle model. In this model the steps of activities are divided into different stages. These stages of work processes are shown in Figure 2.1 with corresponding interconnections.

There are 10 stages in an alarm management lifecycle. The stages are as follows:

1. *Alarm Philosophy* or prior planning stage,
2. *Identification* of points to be monitored,
3. *Rationalization* of the identified points including prioritization and classification,
4. *Detailed Design* of alarms based on rationalized results,
5. *Implementation* of the design,
6. *Operation* as an active mode for an alarm system,

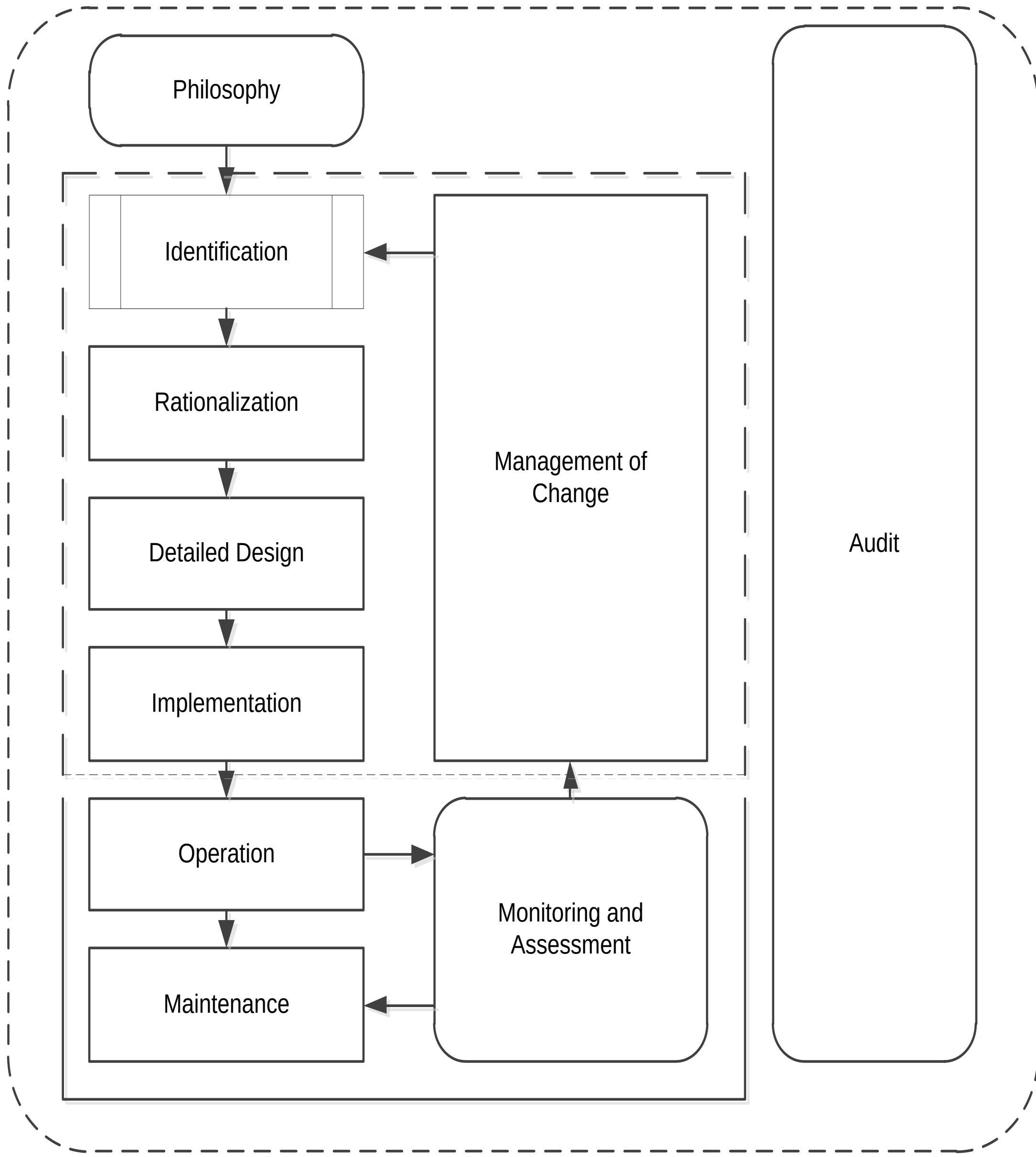


Figure 2.1: Alarm management lifecycle (from ISA 18.2 Standards)

7. *Maintenance* and repairing stage,

8. *Monitoring and Assessment* to achieve desired philosophy,

9. *Management of Change* to continuously update alarm settings, and

10. *Audit* or periodical review.

In this lifecycle model, there are three possible entry points:

- alarm philosophy,

- monitoring and assessment, and

- audit.

For new installations, the entry point is the *alarm philosophy*. This is the planning stage, where the goals and objectives are set and the basic terms such as priorities, performance, and principles are defined. In the *identification* stage, engineers identify points in a plant which may require alarms. These points are then scrutinized and refined through critical analysis and from the understanding of the process in the *rationalization* stage. The modifications and changes made in this stage hold the key for an alarm system to perform optimally. Once the rationalized decisions are made, the *detailed design* is sorted out and implemented in the *implementation* stage. *Operation* and *maintenance* of a system are assessed continuously and taken to the *management of change* stage if necessary. This is the second point of entry which is used as per need basis. Any modifications or changes decided in this stage follow the steps from the *identification* to the *implementation* stage sequentially. *Audit* is the last point of entry where an alarm system is periodically assessed for routine updates and improvements. Work processes started at this point follow the steps from improving alarm philosophy to rest of the steps as connected as shown in the lifecycle model.

The functionality of an alarm system can be enhanced through proper rationalizations. However, during decommissioning a plant, often many hidden rationalization issues are missed or ignored. This may result in high alarm counts and interrelated and sequential alarms.

The proposed alarm flood analysis in this thesis involves analyzing alarm data and investigating patterns in alarm occurrences. Hence, at this point it is necessary to discuss different alarms in typical processes and their uniqueness considered in this study.

2.1.2 Alarm types in the process industry

In the *identification* and *rationalization* stages, various types of alarms are defined as per necessity. For a process, each point is represented with a unique ID called *'tag'*. A tag is a combination of letters and numbers usually indicating the associated unit name, variable type, serial number, and other related information. Each tag consists of several types of alarms. It is decided in the rationalization stage which type is necessary and meaningful to an operator in representing different states of a process. In the ISA 18.2 Standards, several such types are mentioned as follows:

1. Absolute alarms,
2. Deviation alarms,
3. Rate of change alarms,
4. Statistical alarms,
5. Discrepancy alarms,
6. Controller output alarms,
7. Instrument diagnostic alarms,
8. Bad measurement alarms.

Generally, all these alarms can be categorized into two different classes: *continuous alarms* associated with continuously measurable variables and *digital alarms* associated with logical decisions for instruments with control functions.

Continuously measurable variables such as pressure, flow-rate and temperature are associated with a number of alarm limits. Typically, a limit is designed considering several important aspects such as: a) distribution of process variables, b) maximum rate of changes,

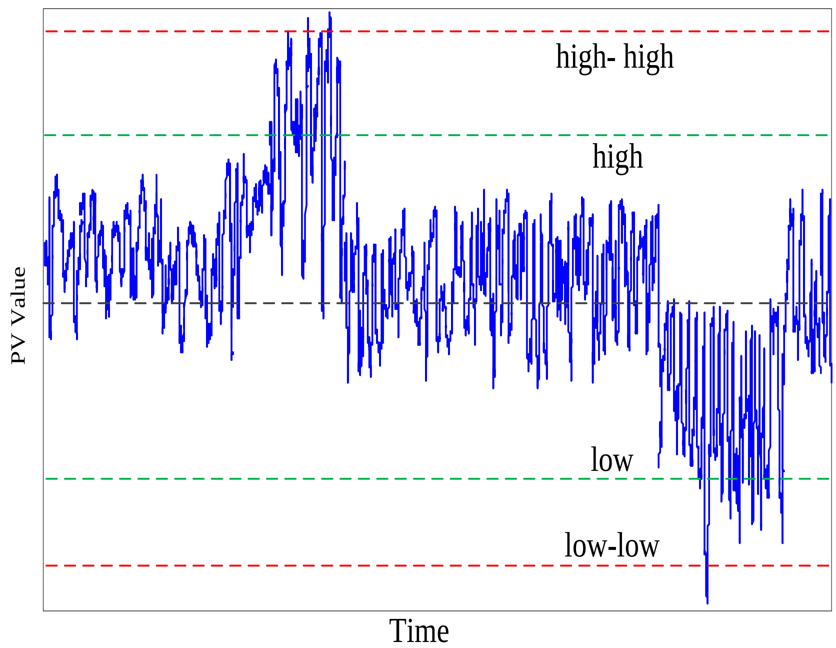


Figure 2.2: Alarm limits and alarm identifiers

c) average response time for an operator, d) limit at which the associated protection operates, e) the amount of risks involved, and f) process condition model [7].

An alarm is annunciated when a variable violates any of the assigned alarm limits. An alarm identifier is assigned to each kind of such violation. Together an identifier and a tag represent the type of fault and the point of occurring. Some of the most commonly used identifiers for measurable process variables are: high (typically identified as PVHI), low (typically identified as PVLO), high-high (typically identified as PVHH) and low-low (typically identified as PVLL). A simple example is shown in Figure 2.2 showing typical alarm limits for a continuously measured variable.

Digital alarms are mostly logical decisions associated with different parts or instruments of a process such as instrument failure, valve malfunction, command failure, instrumentation and measurement failure. For each unique case, a unique identifier is assigned to the corresponding tag decided during a rationalization process.

In the analysis of alarm data, an alarm is considered unique if the combination of the tag and the alarm identifier is unique. A very common approach to represent an alarm event is in the the format of *TAG.ALARM_ID*, indicating both the point at which the fault occurred and the type of the fault. While investigating patterns in an alarm sequence, it is important to clearly define unique alarm events and represent alarm activity via a sequence. Generally, alarm journals keep records of the times of alarm occurrences and clearances along with some other related information. An effective way to represent the alarm activity for a particular span of time is to express it as a timestamped alarm sequence, where each unique alarm event is represented in the format discussed and saved along with its time of annunciation. It consists of two fields: *time* and *alarm*, as a string of time information and alarm symbols respectively.

2.2 Alarm Burst Rates and Alarm Flood Representations

2.2.1 What is an alarm flood?

An alarm flood is the duration where the rate of alarm annunciation is more than the response capability of an operator. In the ISA 18.2 Standards it is stated 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 accordance to this definition, a very common practice is to consider the start of an alarm flood as the time when the alarm count exceeds 10 per 10 minutes of the regular time interval and the finish of an alarm flood as the first time when the alarm rate per 10 minutes falls below the rate of 5 alarm per 10 minutes. Although an alarm rate greater than 10 per 10 minutes indicates a flood, the formation of the flood from alarm count zero to more than 10 is also very important and must be included within the flood period. The initial alarm sequence may contain vital information on the root causes, especially in the case where an alarm flood is caused by interrelated process variables. Therefore, an additional definition on alarm flood duration should be included with the definition of an alarm flood. Complete definitions of alarm floods and flood durations which have been considered throughout this work is as follows:

Definition 1 *An event when an operator experiences more than 10 alarms per 10 minutes is an alarm flood.*

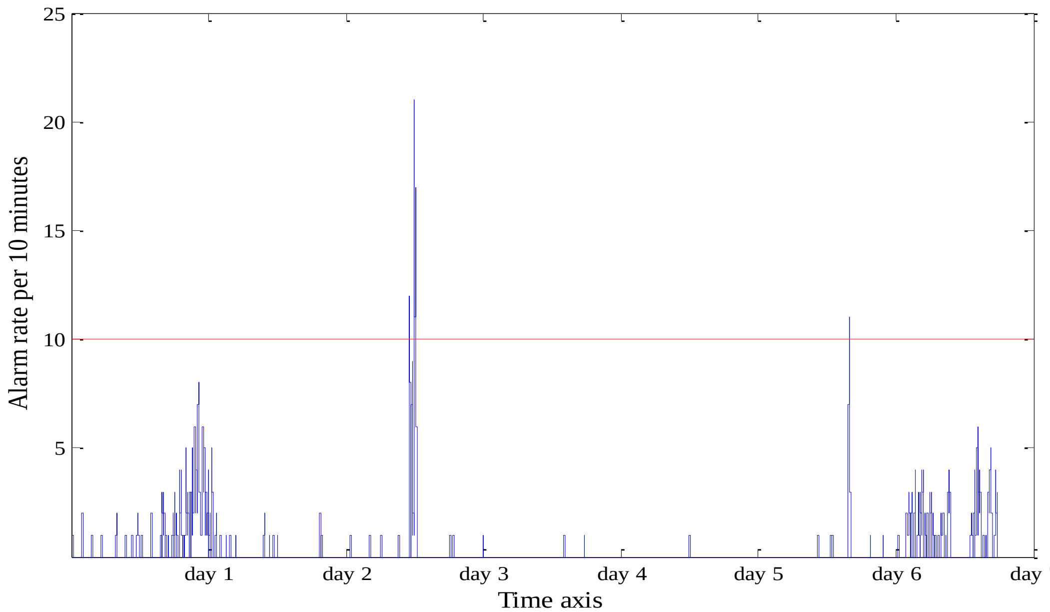


Figure 2.3: Plot for alarm burst rate per 10 minute time slice for a period of one week

Definition 2 *Duration of an alarm flood starts at the time when the alarm rate per 10 minutes starts increasing from zero to greater than 10, and ends at the time when the rate per 10 minutes falls down to zero for the first time after experiencing an alarm flood.*

Given that the definitions of alarm floods and alarm flood durations are given in terms of alarm rate per 10 minutes time slice, an effective way to represent alarm data is through an alarm burst rate plot per 10 minutes time slice. This approach has been first introduced in [37].

An alarm burst rate per T unit time slice is the number of alarms raised within a time window of size T (i.e., number of alarms raised within time $[t-T, t]$, where t is the current time). For a plant, having N number of unique alarm events, the binary sequence $b_i[k]$ for each alarm event b_i, for $i = 1, 2, 3, \cdots, N$, can be defined as follows [19]:

$$b_i[k] = \begin{cases} 1 & \text{alarm at } t;\, T_s + (k-1)\triangle t \leq t < T_s + k\triangle t \\ 0 & \text{otherwise} \end{cases} \qquad (2.1)$$

where, T_s is the time at which the first data sample is collected and $\triangle t$ is the sampling time, typically taken as one second.

A binary sequence of a unique alarm is sparse and very long signal. Often it is expressed and stored as it is shown in Table 2.1.

Table 2.1: Binary sequence

sample	value	sample	value
1-12	0	36-97	0
13	1	98	1
14-34	0	99-207	0
35	1	208	1

The alarm burst rate plot is the plot of function $x[k]$ versus time, where

$$x[k] = \sum_{i=1}^{N} \sum_{n=k-600}^{k} b_i[n] \tag{2.2}$$

for $\triangle t = 1$ second. $x[k]$ is the simple burst rate at a specific time instant representing the number of alarms raised in the 10 minutes interval prior to that instant, and N is the number of unique alarm events.

In Figure 2.3 an alarm burst rate plot is shown for 7 days of real industrial alarm data. Floods can readily be detected from this plot by comparing with the threshold line drawn horizontally at the burst rate of 10 alarms per 10 minutes (assuming there is only one operator involved). Once floods are detected from the burst rate plot, alarm sequences within the flood periods are isolated and expressed as timestamped sequences of the alarm events. Each alarm is stored along with its time of occurrence in two different fields: 'Alarm' and 'Time'.

The two fields 'Alarm' and 'Time' associated with an alarm sequence, are represented as sequences of time information and alarm symbol respectively. For an alarm flood F_m (corresponding to m^{th} recorded alarm flood), the alarm sequence is represented as $\langle S_1^m, S_2^m, \cdots, S_{M_m}^m \rangle$ and the corresponding timestamps represented as $\langle T_1^m, T_2^m, \cdots, T_{M_m}^m \rangle$. Here M_m is the length of the sequence, and S_i^m is the notation corresponding to a symbol representing an unique alarm annunciated at i^{th} position of the m^{th} alarm flood sequence.

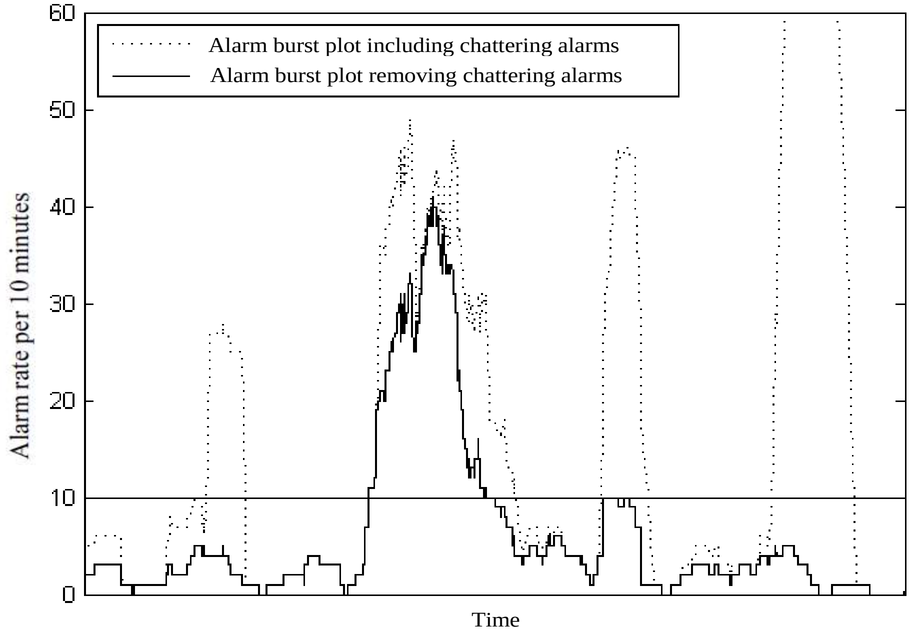


Figure 2.4: Effect of chattering alarms: The dotted line corresponds to the alarm burst rate per 10 minutes including all alarms presented to an operator; the solid line is the burst rate after removing redundant alarm messages for chattering alarm tags.

2.3 Alarm Floods and Chattering

2.3.1 Chattering alarms and their effects

A chattering alarm is an alarm associated with a tag that makes repeated transitions between the normal state and the abnormal state. The most common reason for an alarm to be chattering in nature are a) the presence of noise and b) the corresponding process variable operating at a critical value very close to the alarm limit.

In the process industry, contributions of chattering alarms to the overall alarm count are large. In most cases, operators find large alarm counts which are contributed by a few chattering alarms. Chattering alarms not only increase alarm counts, but are also often interpreted as alarm floods. In Figure 2.4, the alarm burst rate plot of the same process is shown with and without chattering, where the significance of the contributions of chattering alarms can be seen. It can also be seen how few chattering alarms can make the impression of alarm floods with high alarm counts.

In the ISA 18.2 Standards, it is not clearly mentioned if the 10 alarms per 10 minute time slice should include the effect of chattering or not. Usually in industries, the effect of chattering is not removed while calculating different performance metrics. As a result the average number of alarms per flood and the number of alarm floods per reporting period are often found to be high. Moreover, due to the presence of repeating information in alarm messages, alarm sequences may not convey the proper information on the mutual dependencies among the annunciated alarms. Hence removal of nuisance information created by chattering alarms is very important and should be considered for an effective alarm flood analysis.

A comparative study is made between alarm floods detected without eliminating chattering alarm messages and alarm floods detected by eliminating chattering alarms. The results are presented in Tables 2.2 and 2.3.

Table 2.2: Alarm flood metrics (including chattering)

Month	Number of floods	Alarm count per flood
Jan	12	30.9167
Feb	6	55.6667
Mar	10	22.9000
Apr	16	71.6250
May	11	176.0000
Jun	11	225.6364

Table 2.3: Alarm flood metrics (removing chattering)

Month	Number of floods	Alarm count per flood
Jan	7	17.00
Feb	1	12.00
Mar	5	13.40
Apr	8	14.75
May	6	81.83
Jun	6	95.50

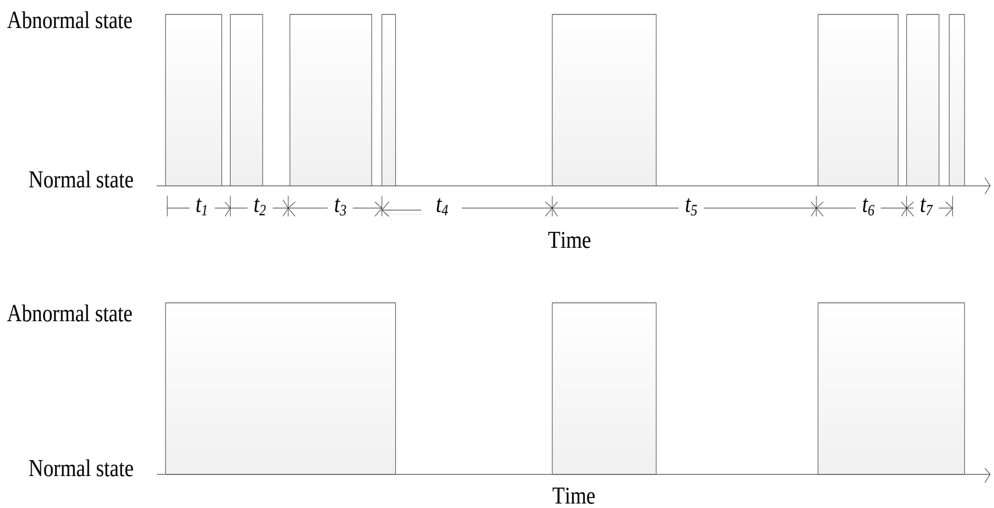


Figure 2.5: Removal of chattering alarms: Consecutive alarms spaced within a time window less than T are combined together. Here $t_1, t_2, t_3 < T$ and combined as a single alarm. Similarly $t_6, t_7 < T$ and combined as another single alarm

2.3.2 Removal of chattering alarms from alarm data

Reduction or elimination of chattering alarms for a process variable is an alarm design problem and is beyond the scope of this thesis. But for post analysis of alarm floods, repeating information of chattering alarms can be easily filtered out. One simple way to do this is by locating clusters of chattering alarms and combine them into single events if their consecutive occurrences are spaced within a narrow time window.

Generally, chattering alarms are lumped together in time with a very short time interval between consecutive occurrences. If the time difference between two consecutive alarm occurrences is less than an allowable limit, the second occurrence can be removed from the corresponding alarm sequence. This eliminates the nuisance information created by chattering tags, losing no vital information on alarm activity. An example of this process is shown in Figure 2.5, where three clusters of alarms can be seen. In two of the clusters, there are multiple numbers of alarms spaced very close to each other due to chattering in the corresponding process variable. In the second plot below, it is shown how such clusters of alarms can be treated as single alarms.

2.4 Summary

In this chapter, basic alarm terminology and systematic alarm management practices have been discussed. A brief discussion on setting of alarms, different types of alarms typically configured in industrial practices, and definition of alarm floods have been presented. Chattering is one of the most undesirable characteristics of alarms commonly found in almost every alarm system in industry. Chattering alarms convey redundant information and are considered as nuisance alarms in the alarm management literature. Removal of chattering alarms is a necessary step for an effective alarm flood analysis. A simple way of removing chattering alarms has been presented in this chapter which is easy to apply for post analysis of alarm data. After removing the chattering alarms, alarm flood data can be isolated from the master alarm database and similarity in patterns of alarm annunciations can be investigated.

Chapter 3

Pattern Analysis of Alarm Floods

3.1 Introduction to Pattern Analysis

In different branches of engineering and science, classification and automatic recognition of patterns are very important. The recent growth in computational power has facilitated the use of massive data classification techniques significantly. Over the last few decades, much research has been carried out on reducing computational cost while maximizing data handling capability. As a result, the literature related to pattern classification is vast and applicable to a variety of sectors in science and engineering.

Typically, a pattern classification or automatic recognition involves three key steps [38]:

1. Data acquisition and preprocessing,
2. Data representation, and
3. Decision making.

Decision making is the classification step, where each pattern is assigned to a particular class. Depending on the data representation, the decision making process may vary. The most common variations are:

- Template matching,
- Structural or syntactic approach,
- Parallel computing approach, and
- Statistical pattern analysis.

Template matching is one of the oldest approaches in pattern recognition. It is mostly used in digital images. In a template matching approach, a template of a query image or query shape is stored and compared with every other test images taking into account various image processing tools, e.g., rotation, translation, correlation, and edges. Typically, it involves high computational costs and is vulnerable in the presence of distortions and deformation of features.

A structural or syntactic approach is used when a hierarchical perspective is desired. In the syntactic approach, a pattern is divided into sub-patterns of different levels. The most elementary level of such sub-patterns is called primitive. A complex pattern is investigated hierarchically in different stages on the basis of the compositions of sub-patterns starting at the primitive level. Although this approach is effective in investigating complex patterns, it suffers from difficulties due to the presence of noise and interference among primitives.

A parallel computing approach is one of the most advanced techniques for pattern analysis. The most useful parallel computing approach is the neural network. As the name indicates, in a neural network a large number of simple computing units are interconnected in a manner similar to the way neurons are connected in a human brain. It can learn to distinguish a class of patterns via training. It is mostly used in bio-informatics and computing science for pattern classifications and automatic recognitions.

In recent years, statistical pattern recognition was found to be quite successful in numerous applications such as computational biology, economics, marketing, artificial intelligence, and large database managements. In a statistical pattern recognition problem, each pattern is represented as a set of d features and viewed as a point in the d dimensional space. The key step is to choose these features so that they can capture the pattern entirely.

For a data set X having d different features, the pattern of the data X can be assigned to one of the N existing classes $c_1, c_2, \cdots, c_N$, using probability theory. In a statistical pattern classification, the probability of the pattern vector of the data X belonging to class c_i is the same as an observation randomly drawn from the class-conditional probability function $P(X|c_i)$. According to the optimal Bayes decision rule, the pattern of X should be assigned to class c_i for which the conditional risk,

$$R(c_i|X) = \sum_{j=1}^{N} L(c_i, c_j) \cdot P(c_j|X) \tag{3.1}$$

is minimum. Here, $L(c_i, c_j)$ is the loss of assigning the pattern to class c_i when the true class is c_j and $P(c_i|X) = \frac{P(X|c_i) \cdot P(c_i)}{P(X)}$. The decision rule of assigning data X to class c_i depends on the choice of the loss function [39]. For example, according to the maximum a posteriori rule (MAP) with a zero-one loss function, the decision rule is as follows:

$$\text{Decision} = \begin{cases} c_i & P(c_i|X) > P(c_j|X) \\ c_j & \text{otherwise} \end{cases} \tag{3.2}$$

However, estimation of a class conditional density is not an easy task. Much research has been published on this regard where class conditional densities are estimated based on training data. Usually, these estimation techniques involve strong assumptions on statistical properties of data which are not very easy to satisfy for practical cases.

In the analysis of alarm flood data presented in this thesis, statistical pattern recognition is considered. Hence, detailed discussions on pattern recognition presented in this chapter are restricted to statistical pattern classification only. Before discussing the classification of alarm flood data, it is important to discuss another property of statistical pattern recognition, leading to supervised and unsupervised learning.

3.2 Supervised and Unsupervised Classifications

There are two modes of operation in a statistical pattern recognition problem:

1. Training, and

2. Testing.

Training is the learning phase, where a pattern classification algorithm learns about different characteristics and their distributions through examples. Examples are commonly called training data sets. Testing is the actual mode of classification. In a testing mode, new data is evaluated and classified based on the pattern distributions leaned in training modes.

For a unknown class conditional density, the learning phase can be either supervised or unsupervised. Supervised learning is the one where each training sample is labeled according to their respective classes, whereas unsupervised learning (also known as unsupervised clustering or clustering) is the one where training samples are not labeled.

In most practical cases, available training samples are expensive or sometimes impossible to label. An unsupervised clustering is suitable for such cases; and hence, it is used in a number of practical applications. Unsupervised clustering is also useful in pattern discovery and feature extraction problems. The motivation for the similarity analysis of alarm floods is to examine patterns in alarm annunciations during flooding periods and compare them with each other. This is a pattern discovery problem, and in this thesis, unsupervised clustering is considered. In the following section, a brief discussion is presented on unsupervised clustering and the associated algorithms.

3.2.1 Unsupervised clustering

Decision boundaries are hypothetical lines separating different classes in data. In an unsupervised clustering, decision boundaries are constructed based on training data. Generally, decision boundaries are identified around a region in a multidimensional feature space, if there exists a high concentration of similar data. Such separated regions are called clusters. Another functional definition of a cluster is often given based on the similarity of data points. From this perspective, a cluster is the collection of data points where the distance between any two data points within the cluster is less than the distance between any data point in the cluster and any data point outside the cluster.

Because training data may reveal clusters with a variety of sizes and shapes, it can be computationally expensive to obtain the optimal decision boundaries. However with different heuristic approaches, it is possible to find clusters with reduced computation, but at the cost of losing optimality. There are many such clustering algorithms which are relatively reliable, consistent and time inexpensive in analyzing large sets of data. In general, clustering algorithms can be categorized to be either *Iterative Partitioning* or *Agglomerative Hierarchical Clustering (AHC).*

3.2.2 Iterative partitioning

Iterative partitioning finds the partition boundaries that minimize the error for assigning each data points to a cluster [38]. One of the most common type of such algorithms is *Squared Error Clustering*.

In squared error clustering, n data points are assigned to K different clusters. If a cluster C_i with a centroid at c_i has n_i data points $x_1^i, x_2^i, \cdots, x_{n_i}^i$ within it, the squared error to be minimized is given as follows:

$$E^2 = \sum_{i=1}^{K} \sum_{l=1}^{n_i} (x_l^i - c_i)^T (x_l^i - c_i). \tag{3.3}$$

A simple iterative algorithm can be used to find boundaries which minimizes the squared error given in equation (3.3). There exists many versions of this algorithm with the same general principle as follows:

Algorithm: Iterative Partitioning

Input: Data set $X = \{x_1, x_2, \cdots, x_n\}$ and the number of clusters K

Output: Clusters $C = \{C_1, C_2, \cdots, C_m\}$, where $m \leq K$

1. Assign arbitrary initial partitions for K clusters;
2. Generate new partitions. Assign each data to the closest cluster center;
3. Compute new c_i for $i = 1, 2, \cdots, K$;
4. Repeat from step 2 if not stabilized;
5. Remove outlier clusters with $n_i <$ threshold;
6. return C.

3.2.3 Agglomerative hierarchical clustering(AHC)

Agglomerative hierarchial clustering is a bottom up approach where each object to cluster is assigned to a separate cluster at the beginning [40]. Distances between two clusters in all possible pairs are computed in each step. A pair of clusters having the least distance

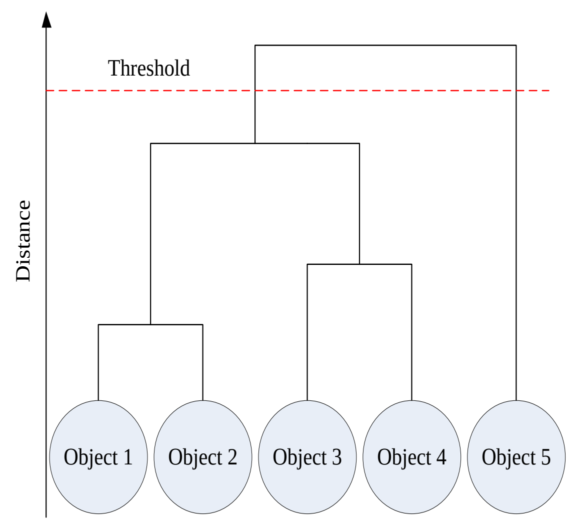


Figure 3.1: Dendrogram of clustering

between them are merged together in each of the following steps. Generally the algorithm stops when all the clusters are merged into a single cluster or when the distance between clusters in a pair (which is least among all) is more than a predefined maximum allowable limit. Typically, an AHC algorithm is represented through a dendrogram as shown in Figure 3.1. In this dendrogram, the vertical axis represents the distance measures between two clusters in a pair and a horizontal line means merging of two clusters. A dendrogram allows one to trace back the history of merging clusters as we move from the bottom to the top. For n objects to merge in a single node, $n-1$ steps are required. The monotonicity condition of merging ensures that the minimum pairwise distance in a step is greater than the minimum pairwise distance in its preceding step; e.g., for minimum pairwise distances $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_{n-1}$ in steps $1, 2, \cdots, n-1$, the following condition holds:

$$\mathcal{D}_1 \leq \mathcal{D}_2 \leq \ldots \leq \mathcal{D}_{n-1}$$

This is the fundamental assumption in agglomerative hierarchical clustering. One of the most advantageous properties of AHC algorithm is, that it is not required to pre-assign the number of clusters. The number of classes can be determined while terminating the algorithm based on a minimum similarity measure [40]. In this thesis, an upper bound on minimum pairwise distances is used to terminate the clustering. For such cases, the

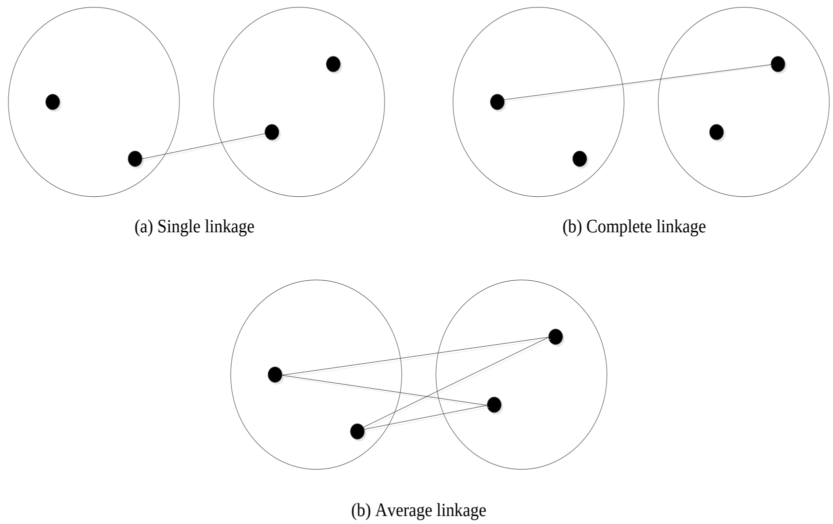


Figure 3.2: Different forms of linkage functions

algorithm stops if the merging pair of clusters in any step is greater than a threshold distance as shown with a dashed horizontal line in Figure 3.1. A simple version of the AHC algorithm is as follows:

Algorithm: AHC

Input: Distance matrix $\mathcal{D}^{K \times K}$ for K objects, threshold τ

Output: Clusters $C = \{C_1, C_2, \cdots, C_m\}$ where $1 \leq m \leq K$

1. Assign each object as a cluster in C
2. Find smallest pairwise distance $\mathcal{D}(i, j)$
3. Merge cluster i & j
4. Update C
5. Update the distance matrix $\mathcal{D}^{K \times K}$ where, $K = K - 1$
6. Go back to step 2 unless $\min\{\mathcal{D}\} > \tau$ OR $K = 1$
7. return C

The AHC algorithm takes different shapes based on the process to update the distance matrix $\mathcal{D}$ in step 5. If we consider that in any step of the algorithm, a new cluster $\mathcal{A}$ is created from merging two different clusters, the pairwise distances associated with the new cluster can be updated using different forms of linkage functions (Figure 3.2). Most common such functions are:

- single linkage,
- complete linkage, and
- average linkage.

A *single linkage function* updates the pairwise distance of the new cluster $\mathcal{A}$ and any other cluster $\mathcal{B}$ with the pairwise distance between the most similar members of the two clusters, i.e.,

$$\mathcal{D}(\mathcal{A}, \mathcal{B}) = \min\{\mathcal{D}(x, y) : x \in \mathcal{A}, y \in \mathcal{B}\}. \tag{3.4}$$

A *complete linkage function* updates the pairwise distance with the distance between the most dissimilar members of cluster $\mathcal{A}$ and cluster $\mathcal{B}$, i.e.,

$$\mathcal{D}(\mathcal{A}, \mathcal{B}) = \max\{\mathcal{D}(x, y) : x \in \mathcal{A}, y \in \mathcal{B}\}. \tag{3.5}$$

In an *average linkage function*, an average of the distances between members of cluster $\mathcal{A}$ and cluster $\mathcal{B}$ is computed and assigned as the new pairwise distance:

$$\mathcal{D}(\mathcal{A}, \mathcal{B}) = \frac{1}{|\mathcal{A}|.|\mathcal{B}|} \sum_{x \in \mathcal{A}} \sum_{y \in \mathcal{B}} \mathcal{D}(x, y), \tag{3.6}$$

where $|\mathcal{A}|$ denotes the number of elements in cluster $\mathcal{A}$.

3.3 Similarity Investigation of Alarm Flood Data

Every process in industry faces a number of alarm floods each year. So generally a question arises: are some of these alarm floods similar to each other in terms of alarm occurrences? It is possible to have a specific sequence of abnormalities in a process that takes place at different points of time due to a common operating condition. This may cause similar alarm floods with similar alarm sequences. These specific sequences of abnormalities are very likely to be the results of closely related process variables feeding one another.

Floods created by similar fault propagation are expected to have similarity in the annunciation of alarms. For floods with longer duration and higher alarm counts, the similarity might not be strictly regular. But if they share identical fault propagation, a good match in alarm subsequences can be expected. This can be investigated through a similarity analysis of alarm flood sequences.

In this thesis, a similarity investigation based on the patterns in alarm annunciation has been performed. In order to find similar alarm floods and group them together, a distance measure is required that can capture the patterns in alarm annunciations. Such a distance measure capturing the similarity in alarm patterns can be used to form a distance matrix and similar alarm floods can be clustered together using the AHC algorithm as described in the previous section.

3.3.1 Problem formulation

To cluster similar alarm floods with the AHC algorithm, pairwise distances need to be calculated for each possible pair of recorded alarm floods. Therefore we define a distance score between two alarm floods in terms of patterns in alarm occurrences as follows:

Given two alarm floods designated as F_m and F_n having alarm sequence $\langle S_1^m, S_2^m, \cdots, S_{M_m}^m \rangle$ with timestamps $\langle T_1^m, T_2^m, \cdots, T_{M_m}^m \rangle$ and alarm sequence $\langle S_1^n, S_2^n, \cdots, S_{M_n}^n \rangle$ with timestamps $\langle T_1^n, T_2^n, \cdots, T_{M_n}^n \rangle$, respectively, a distance score $\mathcal{D}(F_m, F_n)$ is to be calculated between the two flood sequences in terms of patterns in alarm occurrences. The distance $\mathcal{D}(F_m, F_n)$ must capture the dissimilarity in alarm patterns and satisfy the standard properties of dissimilarity measures given as follows [41]:

- Positivity: $\mathcal{D}(F_m, F_n) \geq 0$
- Symmetry: $\mathcal{D}(F_m, F_n) = \mathcal{D}(F_n, F_m)$
- Minimality: $\mathcal{D}(F_m, F_m) \leq \mathcal{D}(F_m, F_n), \quad \forall\, m, n$

Here, the equality in *positivity* holds for two exactly same alarm flood sequences in terms of alarm annunciation patterns. *Minimality* is to ensure that a distance calculated for two different alarm floods is always greater than or equal to the distance calculated for any of the alarm floods to itself. *Symmetry* is a common property in similarity measure. It indicates that a distance between two floods should be independent of their order. One important thing to notice is that this distance measure is based on patterns in alarm sequences and unlike standard geometric distance measures, it is not required to satisfy the *triangular inequality*.

3.3.2 Distance calculation

As alarm flood sequences are timestamped event sequences, a direct measure of pairwise distance is not feasible. A general idea is to fix a feature space $\mathcal{F}^l$ where l number of features capturing patterns of a sequence can be projected and a local distance $\mathcal{D}$ can be defined as follows:

$$\mathcal{D} : \mathcal{F}^l \times \mathcal{F}^l \rightarrow \mathbb{R}_+. \tag{3.7}$$

Typically $\mathcal{D}(F_m, F_n)$ is small if the corresponding sequences are similar and large if they are different. The key step is to select an appropriate feature space. Once these pairwise local distances between all possible pairs of alarm floods are computed, the floods can be clustered using the agglomerative hierarchial clustering algorithm.

3.4 Summary

In this chapter, a brief overview on pattern recognition and classification has been presented. A detailed discussion on unsupervised statistical pattern classification has been given along with the algorithms to find groups in an unknown set of data. Unsupervised classification techniques are very useful tools in analyzing data for feature extraction. To apply unsupervised classification in alarm flood data, a distance measure is needed to be defined that can capture the patterns in alarm annunciations. In the next chapter some suitable notions of such distance measures are introduced which are later used to cluster recorded alarm floods from a real industrial plant.

Chapter 4

Distance Measures

4.1 Introduction

This chapter discusses different distance measures considered in the similarity analysis of alarm floods. Different features are considered while defining distance scores on alarm patterns. As discussed in Chapter 2, an alarm flood data set consists of two fields: *Time* and *Alarm*. For simplicity, the time differences among annunciated alarms are ignored in the study presented in this thesis. Another reason for not considering time differences is that, a process generally operates in different operating conditions in different times with different production targets; a similar train in fault propagation might not have similarity in timestamps of annunciated alarms.

One of the biggest concerns in a pattern analysis is the cost of computation. Fortunately, not every alarm flood shares all identical alarms. Hence, it is not required to measure pairwise distances for pairs in which, the two alarm floods do not have a significant portion of alarms in common. Based on this observation, a pre-assignment of pairwise distance is considered. Pairwise distances are calculated only for those alarm floods which share many common alarms and they are mainly based on alarm orders in corresponding sequences. Three such distance measures are considered in this work: a) a distance based on frequencies of consecutive alarms, b) a distance based on relative order of alarms in a sequences, and c) a distance based on dynamic time warping sequence alignment.

4.2 Filtering Alarm Flood Pairs Based on Common Alarm Events

An alarm flood caused by interrelated process variables is expected to be composed of a specific set of alarms. Two different flood sequences can be similar only if they have a good percentage of alarm in common in their alarm sequences. Typically, the number of unique alarms in a process are higher than the average number of alarms per flood. So a pair of alarm floods which have a negligible number of alarms in common are never expected to be similar. A pattern based distance calculation on such pairs are unnecessary. Such pairs can be pre-assigned a high distance to reduce computational costs significantly.

A simple investigation can be made prior to similarity calculations in order to find if a pair of alarm floods have enough alarms in common or not. An alarm flood F_m can be represented with a vector $\mathcal{E}_m^{1\times N}$ for N unique alarms associated with the plant under study as follows:

$$\mathcal{E}_m = [E_1, E_2, E_3, \cdots, E_N] \quad \text{where,}$$

$$E_i = \begin{cases} 1 & i^{th} \text{ unique alarm is present at the sequence} \\ 0 & \text{otherwise} \end{cases}. \tag{4.1}$$

Once two alarm floods in a pair F_m and F_n are expressed in terms of such vectors, a suitable binary distance can be calculated. There exist many binary distance measures to find similarity in attributes between binary sequences; e.g., those defined in

- Dice (1945),
- Jaccard (1908),
- Sorensen (1948),
- Rifqi et al. (2000),
- Sneath and Sokal (1973).

In this work, a *Jaccard* distance is calculated between each pair of alarm floods to pre-assign a maximum normalized distance 1 depending on alarm event dissimilarity. A

Jaccard distance is a very commonly used type 1 distance measure for binary data. In a type 1 distance measure, the measure depends only on the attributes present in either of the objects and independent of the attributes absent in both the objects. A Jaccard distance between two data $\mathcal{E}_m$ and $\mathcal{E}_n$ with N binary attributes is given as follows:

$$J(\mathcal{E}_m, \mathcal{E}_n) = \frac{b+c}{a+b+c}; \tag{4.2}$$

where

- a is the number of attributes common to both objects, i.e., $|\mathcal{E}_m \bigcap \mathcal{E}_n|$
- b is the number of attributes present in $\mathcal{E}_m$ but not in $\mathcal{E}_n$, i.e., $|\mathcal{E}_m \backslash \mathcal{E}_n|$
- c is the number of attributes present in $\mathcal{E}_n$ but not in $\mathcal{E}_m$, i.e., $|\mathcal{E}_n \backslash \mathcal{E}_m|$

For a pair of alarm floods, a Jaccard distance between the two corresponding vectors close to zero implies that the two sequences have almost all the same alarms and very likely to be caused by similar dynamics among different process variables. A further investigation on the patterns of such alarm sequences can be made to confirm similarity in alarm annunciation patterns.

4.3 Frequency of Consecutive Alarms

In a process, interrelated alarms follow causal relationships among each other. For constant alarm thresholds and steady operating conditions, it is expected to find similar patterns in terms of consecutive alarms. An alarm sequence of a flood caused by tightly related process variables is expected to be composed in a cause-effect manner, i.e., for a sequence $\langle S_1^m, S_2^m, S_3^m, \ldots, S_{M1}^m \rangle$, it can be speculated that S_2^m might have been caused by S_1^m, S_3^m might have been caused by S_2^m and/or S_1^m and so on. For such cases, the alarm sequence can be assumed to have generated from a Markov chain.

A first order Markov chain is a sequence of random variables, where the probability that S_i^m takes a particular value depends only on the preceding variable S_{i-1}^m. A first order Markov chain can be completely specified by $N \times N$ transition probabilities: $p(x|y)$ where, x and y are any two alarms associated with the plant that has N unique alarms. Estimation of

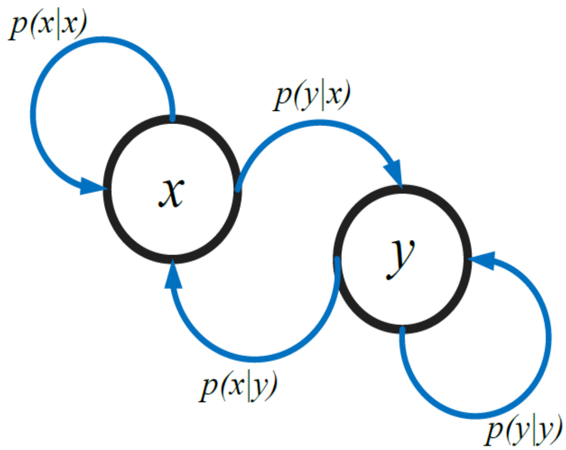


Figure 4.1: First order Markov chain with two states

.

these transition probabilities requires large and consistent alarm flood sequences, which are practically impossible to obtain. An analogy to the concept is to calculate the frequencies of consecutive alarms within a given sequence. Thus a flood sequence can be expressed into a set of distinct features mapped to a feature space $\mathcal{F}^{N^2}$, with N^2 features expressed as a matrix $\mathcal{P}$ as follows:

$$\mathcal{P} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ f_{N1} & f_{N2} & \cdots & f_{NN} \end{bmatrix}.$$

Here, f_{ij} is the frequency of alarm i coming immediately after alarm j within a specific flood sequence and N is the total number of unique alarms.

For two different alarm flood sequences $\langle S_1^m, S_2^m, \cdots, S_{M_m}^m \rangle$ and $\langle S_1^n, S_2^n, \cdots, S_{M_n}^n \rangle$, an Euclidean distance can be calculated and normalized as the distance score $\mathcal{D}_1(F_m, F_n)$ between the two floods as follows:

$$\mathcal{D}_1(F_m, F_n) = \left(\sum \sum (\mathcal{P}_m - \mathcal{P}_n)^2 \right)^{\frac{1}{2}}. \tag{4.3}$$

4.4 Distance Measure Based on Relative Alarm Occurrences

For an alarm flood sequence, $\langle S_1^m, S_2^m, \cdots, S_{M_m}^m \rangle$ resulting due to interrelated alarms, the alarm S_3^m might have been caused by either S_2^m or S_1^m. Frequencies of consecutive appearance would not be able to capture the dependency if S_3^m is caused by S_1^m and S_2^m is just a random alarm. For such cases, a relative measure of alarm occurrence could be more effective. Alarm floods caused by similar alarms may also contain many unrelated alarms. Hence, the common alarms in a pair of alarm floods are only necessary to consider while comparing patterns of alarm occurrences. Let us denote the alarm sequences of two floods F_m and F_n with common alarms only as $\hat{S}^m$ and $\hat{S}^n$, and define a matrix $\mathcal{R}_m$ capturing the relative occurrences of the common alarms in $\hat{S}^m$ for the pair of floods F_m and F_n, such that:

$$\mathcal{R}_m = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ r_{N1} & r_{N2} & \cdots & r_{NN} \end{bmatrix}$$

where,

$$r_{ij} = \begin{cases} +1 & \text{if alarm } i \text{ first appeared after alarm } j\text{'s first appearance in alarm sequence } \hat{S}_m \\ -1 & \text{otherwise} \end{cases} \tag{4.4}$$

For two alarm flood sequences F_m and F_n, a logical similarity investigation between $\mathcal{R}_m$ and $\mathcal{R}_n$ can reflect the measure of similarity in alarm appearances. It can be calculated in a similar way as for the Jaccard index for binary data as follows:

$$\mathcal{D}_2(F_m, F_n) = \frac{M_{(1,-1)} + M_{(1,0)} + M_{(-1,1)} + M_{(-1,0)} + M_{(0,1)} + M_{(0,-1)}}{M_{(1,1)} + M_{(-1,-1)} + M_{(1,-1)} + M_{(1,0)} + M_{(-1,1)} + M_{(-1,0)} + M_{(0,1)} + M_{(0,-1)}}. \tag{4.5}$$

Here, $M_{(a,b)}$ denotes the total number of attributes where $\mathcal{R}_m$ is equal to a and $\mathcal{R}_n$ is equal to b.

4.5 Dynamic Time Warping

Dynamic Time Warping (DTW) is a nonlinear time alignment method used in aligning time dependent sequences. It has been found to be very useful in the field of pattern recognition of real valued data in various sectors such as speech recognitions, electrocardiograph, scientific database, video, and graphics. Here in this work, it has been used for pattern analysis of alarm floods.

Unlike other techniques in pattern recognitions, continuity in a sequence is not regarded as a very crucial property. It is rather used for sequences and signals which have different lengths. A very good example of such a signal is speech. The duration of utterance for a particular word varies from person to person and from time to time. To compare such signals, alignment is a necessary step. Time warping allows one to stretch and compress such time series data along the time axis and map one element of a sequence to one or many elements of the other sequence. Thus two sequences can be compared effectively even if they have different lengths.

For time sequences of different lengths, the simplest solution of time alignment is a linear time alignment. For example, let us consider two time sequences $X \Rightarrow \langle x_1, x_2, \cdots, x_m \rangle$ and $Y \Rightarrow \langle y_1, y_2, \cdots, y_n \rangle$ where $m \neq n$. We use i_x to denote the time indices of X and i_y to denote time indices of Y. A linear time alignment distance measure between the two sequences with the direction of time normalization towards X can be given as:

$$\mathcal{D}(X, Y) = \sum_{i_x=1}^{m} \mathcal{D}(x_{i_x}, y_{i_y}), \tag{4.6}$$

where $\mathcal{D}(x_{i_x}, y_{i_y})$ is the predefined distance for the pair of elements x_{i_x} and y_{i_y}, and i_x and i_y satisfy $i_y = \frac{n}{m} \times i_x$ subjected to a suitable round-off rule (as $i_x, i_y \in \mathbb{Z}$). In a linear time alignment problem, the variation in a sequence is assumed to be proportional to the length of the sequence. For most of the cases, this is a hard condition to satisfy. A solution to this problem is to introduce a nonlinear time alignment by warping the time axis as per need basis. Here, let us introduce a warping function w, which warps the time indices i_x and i_y in a nonlinear way as follows:

$$i_x = w_x(k) \qquad k = 1, 2, \cdots, K, \tag{4.7a}$$

$$i_y = w_y(k) \qquad k = 1, 2, \cdots, K. \tag{4.7b}$$

In this case a new distance measure for time warped sequences can be defined as:

$$\mathcal{D}_w(X, Y) = \sum_{k=1}^{K} \mathcal{D}(x_{w_x(k)}, y_{w_y(k)}), \tag{4.8}$$

Here, $x_{w_x(k)}$ denote samples of the sequence X with a nonlinearly increasing time indices. Figure 4.2 shows an example of linear and nonlinear time alignment. Here, it can be seen

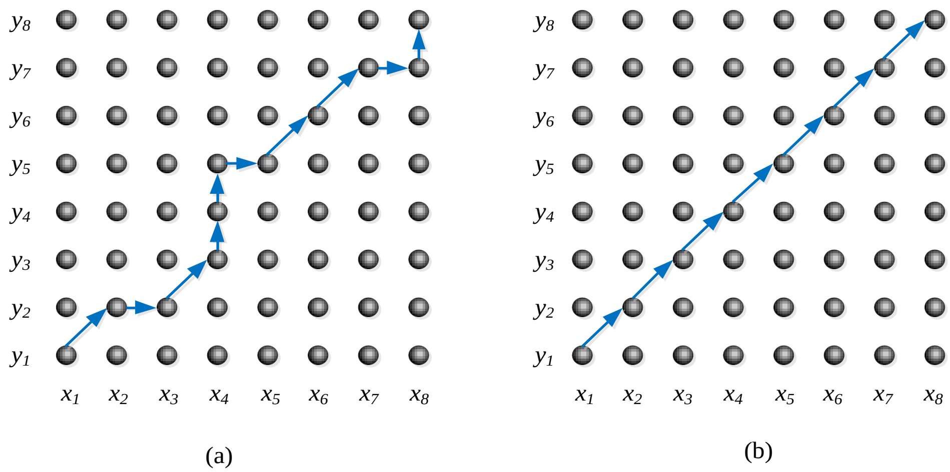


Figure 4.2: (a) Nonlinear time normalization (b) linear time normalization

.

that in a linear time alignment, time index in a sequence increases linearly and the alignment is along the diagonal. Whereas, in a nonlinear time alignment, it increases differently for different elements and the alignment is not strictly bound along the diagonal.

The nonlinear warping path shown in Figure 4.2, is not unique in general. Many such paths can be found for nonlinear alignment of two time series sequences. The most common practice is to find an optimal warping path that incurs a minimum time alignment distance. However, the objective to find a path with minimum distance alone is not enough for a unique solution. Many other constraints are often introduced to reduce pathological paths depending on the nature of the sequences to align.

Finding an optimal path is a tedious job and computationally expensive. A clever way to find a solution to this kind of problem is to solve it sequentially. The method to solve a complex problem sequentially is called *Dynamic Programming*.

4.6 Time Warp Distance Calculation

4.6.1 Introduction to dynamic programming

Dynamic programming is one of the most widely used tools in solving sequential decision problems. In the optimal path problem mentioned earlier, dynamic programming can find solutions with a considerably less computation [42]. For example, let us consider N different

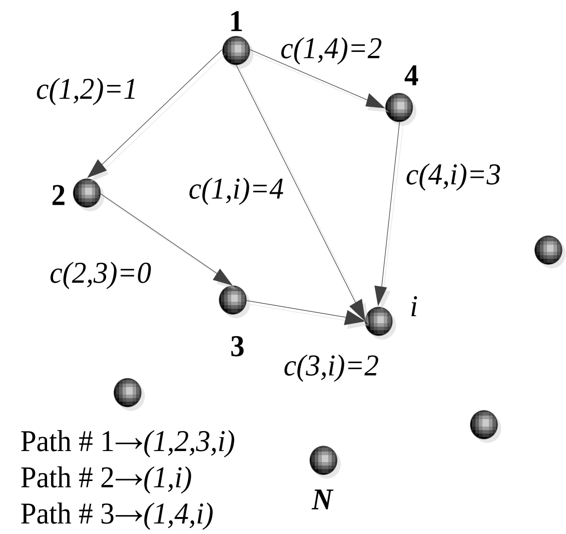


Figure 4.3: 3 paths from point 1 to i are shown. The associated costs are (i) for path 1, $c(1,2) + c(2,3) + c(3,i) = 3$ (ii) for path 2, $c(1,i) = 4$ (iii) for path 3, $c(1,4) + c(4,i) = 5$. The path incurring minimum cost among the three is path 1.

points given as in Figure 4.3 with nonnegative costs of moving from one point to another. An optimal path starting from a specific point to another is defined as a set of moves that produce a least cost among all other possibilities. In dynamic programming, such a decision is broken into smaller subproblems subject to a fixed set of constraints and solved sequentially [43]. The optimality is held in each subproblem and maintained till the end of the solution. This principle of optimality was first introduced by Bellman in [42], commonly known as the *Bellman's Principle of Optimality*. There, it is stated as:

> "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."

In Figure 4.3, N different points are labeled from 1 to N. Let us consider finding an optimal path from point 1 to point j. We define the associated cost for moving from any point i to any point j in one step as $c(i,j)$ and the minimum cost of moving from point i to any point j in one or more steps, as $\varphi(i,j)$. According to the principle of optimality, if i is the point immediately prior to point j then the minimum cost $\varphi(1,j)$ is the sum of $\varphi(1,i)$ (in one or more steps) and $c(i,j)$ (in one step).

$$\varphi(1,j) = c(i,j) + \varphi(1,i). \tag{4.9}$$

So, in general the minimum cost in moving from any point i to j via any point (there should exist one) can be given as follows:

$$\varphi(i,j) = \min_k\{\varphi(i,k) + \varphi(k,j)\}. \tag{4.10}$$

In dynamic programming, this problem of finding the minimum cost in moving from one point to another is divided into smaller subproblems. Each subproblem is calculated sequentially starting from the subproblem that finds the optimal path in a single move to the subproblem that finds the optimal path in a maximum allowable number of moves. In each step, the result of the previous subproblem can be used. For example, in dynamic programming, the minimum cost in moving from i to j in L number of moves is calculated sequentially as follows:

$$\begin{aligned}
\varphi_1(i,j) =& c(i,j) && \forall j = 1,2,\cdots,N, \\
\varphi_2(i,j) =& \min_k\{\varphi_1(i,k) + c(k,j)\}, && \forall k = 1,2,\cdots,N, \quad \forall j = 1,2,\cdots,N \\
\varphi_3(i,j) =& \min_k\{\varphi_2(i,k) + c(k,j)\}, && \forall k = 1,2,\cdots,N, \quad \forall j = 1,2,\cdots,N \\
&\vdots \\
\varphi_L(i,j) =& \min_k\{\varphi_{L-1}(i,k) + c(k,j)\}, && \forall k = 1,2,\cdots,N, \quad \forall j = 1,2,\cdots,N
\end{aligned}$$

Here, φ_p indicates the minimum cost of moving in p number of moves. Finally, the minimum cost in moving from point i to point j can be found as follows:

$$\varphi(i,j) = \min_{1\le l\le L}\{\varphi_l(i,j)\} \tag{4.11}$$

where L is the maximum number of moves allowed.

Based on the number of moves in an optimal path, sequential decision problems can be categorized into two different classes:

- Synchronous sequential decision problems, and
- Asynchronous sequential decision problems.

If the number of moves in an optimal path is not specified, it is called an asynchronous decision problem, e.g, the example just discussed. Generally, asynchronous sequential decision problems are computationally more expensive in comparison with to synchronous sequential decision problems. In a synchronous sequential decision problem, the number of moves in an optimal path is fixed and often it is represented with a trellis structure [43]. For N number of points with L fixed moves, the computation in a synchronous sequential decision problem is in the order of $N \times L$.

In an optimal alarm sequence mapping, one needs to find a path that incurs least number of mismatches in mapping. This is a similar problem as the problem discussed in the example; and dynamic programming can be used for fast and reliable calculation.

4.6.2 Optimal mapping of alarm sequences using DTW

Let us consider two alarm sequences $S^m \Rightarrow \langle S_1^m, S_2^m, \cdots, S_{M_m}^m \rangle$ and $S^n \Rightarrow \langle S_1^n, S_2^n, \cdots, S_{M_n}^n \rangle$ associated with alarm floods F_m and F_n respectively. We define a local cost $c(S_i^m, S_j^n)$ for each possible pair of elements mapped together from the two sequences. Typically, for real valued signal, such a local cost is defined as any suitable $\mathcal{L}_p$ distances [44]. As alarm sequences are not real valued sequences, a local cost $c(S_i^m, S_j^n)$ is defined as a zero-one function as follows.

$$c(S_i^m, S_j^n) = \begin{cases} 0 & \text{if } S_i^m \text{ and } S_j^n \text{ are the same} \\ 1 & \text{otherwise} \end{cases} . \tag{4.12}$$

For the two alarm sequences S^m and S^n, a cost matrix $C(S^m, S^n) \in [1,\, 0]^{M_m \times M_n}$ can be formed from the corresponding local costs such that, $C(i,j) = c(S_i^m, S_j^n)$. The optimal mapping between the two sequences is the optimal warping path along the cost matrix C, composed of different points in it incurring minimum cost for the two entire sequences.

However, Optimal path with a minimum cost is not unique in general and many of the optimal paths might not be meaningful at all. For example, there may exist 2 or 3 alarms in one flood sequence mapped to another entire flood sequence. Such an optimal path in a cost matrix is called a pathological warping path. To have a meaningful alignment between two sequences, additional constraints can be introduced [43]. Constraints which are considered in this work for the analysis of alarm flood data are:

1. Boundary condition,
2. Monotonicity constraints, and
3. Local continuity constraints.

Boundary conditions

A boundary condition or endpoint constraint is a constraint to ensure the total comparison of the two sequences under consideration. As the lengths of alarms flood sequences are not necessary to be the same, this restriction makes sure that the warping path starts at the beginning of both alarm sequences and ends at the end of both sequences. It is often stated in terms warping functions as follows:

$$w_m(1) = 1, \;\; w_n(1) = 1, \tag{4.13a}$$

$$w_m(K) = M_m, \;\; w_n(K) = M_n. \tag{4.13b}$$

Monotonicity constraints

A monotonicity constraint ensures proper considerations of the temporal order of the alarms in a sequence. Similar fault propagation in a process should raise alarms in a similar order. For alarm sequences S^m and S^n with warping functions w_m and w_n respectively, monotonicity constraints are expressed as follows:

$$w_m(k+1) \geq w_m(k), \tag{4.14a}$$

$$w_n(k+1) \geq w_n(k). \tag{4.14b}$$

It restricts time reversal in mapping and guarantees the warping path to always have positive slopes in time normalization.

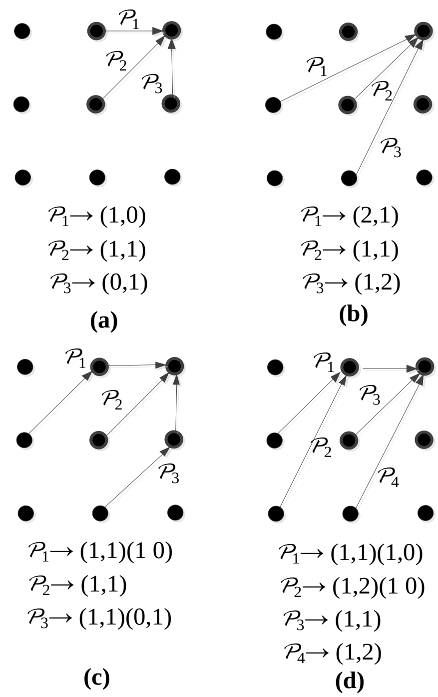


Figure 4.4: Local continuity constraints.

Local continuity constraints

Local continuity constraint also implies the monotonicity constraints within it. Yet it is mentioned separately because of its importance. While warping in time, a local continuity constraint defines the allowable warping in a single step. Often, local continuity constraints are expressed in terms of allowable paths or coordinate changes in a single step.

There are many forms of local continuity constraints. Most of these are heuristically derived for a specific application. Some of the most commonly used forms are given in Figure 4.4. The simplest and most commonly used among all is the one at Figure 4.4(a), where the allowable path changes are:

$$\mathcal{P}_1 \to (1,0), \tag{4.15a}$$

$$\mathcal{P}_2 \to (1,1), \tag{4.15b}$$

$$\mathcal{P}_3 \to (0,1). \tag{4.15c}$$

It is also expressed in terms of warping functions as follows:

$$w_m(k+1) - w_m(k) \leq 1 \tag{4.16a}$$

$$w_n(k+1) - w_n(k) \leq 1 \tag{4.16b}$$

With all the constraints discussed, it is possible to find a unique optimal warping path along the cost matrix starting from the $C(1,1)$ to $C(M_m, M_n)$. A warping path can also be expressed with changes in coordinates starting from $C(1,1)$ to $C(M_m, M_n)$, e.g.,

$$\mathcal{P}_{warp} \rightarrow (p_1, q_1)(p_2, q_2)(p_3, q_3) \cdots (p_T, q_T) \tag{4.17}$$

For such a warping path $\mathcal{P}_{warp}$ with associated warping functions w_m and w_n, the following conditions are held:

$$w_m(k) = \sum_{i=1}^{k} p_i \tag{4.18a}$$

$$w_n(k) = \sum_{i=1}^{k} q_i \tag{4.18b}$$

$$\sum_{k=1}^{T} p_k = M_m \tag{4.18c}$$

$$\sum_{k=1}^{T} q_k = M_n \tag{4.18d}$$

As mentioned earlier, the raw search of an optimal path could be computationally expensive. A raw search where one tests every possible warping path, has exponential computational complexity in the length of the two sequences. This is similar to the problem discussed in Section 4.6.1, and an optimal path along the cost matric C can be found using dynamic programming as discussed in the following section.

4.6.3 Calculation of DTW distance

A dynamic time warping distance $\mathcal{D}_{tw}(S^m, S^n)$ between two alarm sequences S^m and S^n, subjected to (i) boundary conditions (equation 4.13), (ii) monotonicity constraints (equation 4.14) and (iii) a local continuity constraints (equation 4.15), can be given as:

$$\mathcal{D}_{tw}(S^m, S^n) = c(S^m_1, S^n_1) + \min\{\mathcal{D}_{tw}(S^m, S^n_{2:M_n}), \mathcal{D}_{tw}(S^m_{2:M_m}, S^n), \mathcal{D}_{tw}(S^m_{2:M_m}, S^n_{2:M_n})\}, \tag{4.19}$$

where $S^m_{p:q}$ denotes the subsequence of S^m starting from the p^{th} element to the q^{th} element. To solve this problem sequentially, let us define a recurrence relation $\gamma(S^m_i, S^n_j)$ such that:

$$\gamma(i,j) = \mathcal{D}_{tw}(S^m_{1:i}, S^n_{1:j}), \tag{4.20}$$

which can be computed efficiently from the following relation:

$$\gamma(S^m_i, S^n_j) = c(S^m_i, S^n_j) + \min\{\gamma(S^m_i, S^n_{j-1}), \gamma(S^m_{i-1}, S^n_j), \gamma(S^m_{i-1}, S^n_{j-1})\}. \tag{4.21}$$

A cumulative distance table Γ of size $M_m \times M_n$ can be built from the recurrence relations such that, $\Gamma(i,j) = \gamma(S^m_i, S^n_j)$. The optimal mapping between the sequences can be obtained through tracing backwards in Γ starting from $\Gamma(M_m, M_n)$ to $\Gamma(1,1)$. In each step, an allowable previous cell ($\mathcal{P}_1 \rightarrow (-1,0)$, $\mathcal{P}_2 \rightarrow (-1,-1)$ and $\mathcal{P}_3 \rightarrow (0,-1)$) with the lowest cumulative distance is selected and the elements indicated by the indices of the selected cell are mapped together [45]. The algorithm to find such mapping is shown as follows:

Algorithm: Optimal Warping Path

Input: Cumulative distance table Γ

Output: Optimal warping path $\mathcal{P}_{warp} \rightarrow (p_1, q_1)(p_2, q_2) \cdots (p_T, q_T)$

1. initiate with $(p_T, q_T) = (M_m, M_n)$ for $i = T$
2. continue till $(p_i, q_i) = (1,1)$ for $i = 1$
3. update $(p_{i-1}, q_{i-1}) = \begin{cases} (1, q_i - 1), & \text{if } p_i = 1 \\ (p_i - 1, 1), & \text{if } q_i = 1 \\ \arg\min\{\Gamma(p_i - 1, q_i - 1), \Gamma(p_i - 1, q_i), \Gamma(p_i, q_i - 1)\} & \text{otherwise} \end{cases}$
4. $(p_{i-1}, q_{i-1}) \rightarrow \mathcal{P}_{warp}$
5. return $\mathcal{P}_{warp}$

An example of an optimal mapping that generates a minimum cost for the two sequences $\langle a, b, c, d, e\rangle$ and $\langle b, c, e\rangle$ is shown in Figure 4.5.

4.7 DTW Based Distance Measure for Alarms Floods

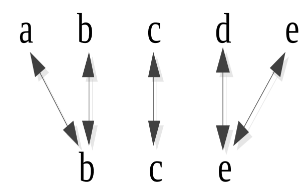
Dynamic time warp based distances are calculated to investigate whether or not the common alarms in two different alarm flood sequences follow similar order in their annunciations.

	a	b	c	d	e
b	1	0	1	1	1
c	1	1	0	1	1
e	1	1	1	1	0

(a)

	a	b	c	d	e
b	$\underline{1}$	$\underline{1}$	2	3	4
c	2	2	$\underline{1}$	$\underline{2}$	3
e	3	3	2	2	$\underline{2}$

(b)



(c)

Figure 4.5: Dynamic time warping between two sequences: (a) cost matrix, (b) cumulative distance table, (c) optimal mapping between the two sequences.

For two alarm floods denoted by F_m and F_n, only the common alarms in alarm sequences are kept and the rests are removed while finding the optimal mapping. For the two alarm sequences $\hat{S}^m$ and $\hat{S}^n$, the cost matrix $C(\hat{S}^m, \hat{S}^n)$ and the corresponding cumulative distance table Γ can be easily calculated as discussed in the previous section. Using the algorithm to find the optimal path along the cost matrix C, one can obtain the optimal mapping of the two sequences. The last value in the cumulative distance table denotes the DTW distance $\mathcal{D}_{tw}(\hat{S}^m, \hat{S}^n)$ and it can be normalized over the total number of points in the optimal path. This measure signifies the number of mapping that maps similar alarms in the two sequences over the total number of mapping used in the alignment.

For an optimal path given as $\mathcal{P}_{warp} \rightarrow (p_1, q_1)(p_2, q_2)\cdots(p_T, q_T)$, the DTW based distance between the two alarm sequences is:

$$\begin{aligned} \mathcal{D}_3(F_m, F_n) &= \tfrac{1}{T} \times \mathcal{D}_{tw}(\hat{S}^m, \hat{S}^n) \\ &= \tfrac{1}{T} \times \Gamma(|\hat{S}^m|, |\hat{S}^n|). \end{aligned} \tag{4.22}$$

4.8 Summary

In this chapter, different distance measures based on patterns of alarm annunciations have been discussed. Distance measures have been used to calculated pairwise dissimilarity between two flood sequences. In some of the pairs of the recorded floods, it is not required to investigate pattern similarity in alarm occurrences as the corresponding flood sequences do not share adequate number of common alarm events. Such pairs can be assigned a high distance based on the Jaccard distance calculated on common alarm attributes. Once all the pairwise distances for recorded alarm floods are available, similar alarm floods can be clustered into groups using the AHC algorithm.

Chapter 5

Case Study

5.1 Unit Overview

The proposed similarity analysis of alarm flood data is carried out on a real industrial alarm data set. The alarm data was taken from an oil hydro treater unit that reduces sulphur content from the up stream feed. The major equipment involved with the unit are: furnace, compressor, pumps, amine scrubber towers, separation drums and stripper tower. There are a total of over 1,300 tags and over 1,700 unique alarms associated with the process with a well defined priority breakdown.

Under normal operating conditions, the unit has excellent alarm statistics. But the unit has had a good number of alarm floods and many of them are suspected to be caused by highly interrelated alarms. Although a few of such interdependent alarms can be explained by the process engineers from their understanding of the process, there is a likelihood of many other undiscovered interrelations. The proposed similarity analysis is independent of process knowledge and requires only alarm information to find such hidden interrelations. A nine months worth of alarm data from the unit was analyzed. Similar alarm floods are clustered into groups based on patterns of alarm annunciation. A brief description of the study is presented in the following section.

5.2 Analysis

Step 1

Typically, alarm journals keep records of many alarm related information with timestamps such as tag names, identifiers, message types, alarm thresholds, etc. The analysis of alarm floods proposed in this work requires only timestamped alarm information where each alarm event is composed of a unique combination of a process tag and an alarm identifier. In the first step, alarm data is extracted from the alarm journal as timestamped alarm sequences. Each of the unique alarm is assigned a specific index for ease of analysis.

Step 2

In step 2, the effect of chattering alarms is removed. The same alarms raised within a time window of 1 minute are considered as repeating alarms and combined into a single event. A time window of 1 minute is chosen as it is generally expected that an operator can take care of only one alarm every minute under normal conditions. It has been seen that after the removal of the redundant information created by chattering alarms, both the number of alarm floods and the alarm count in each alarm flood were reduced significantly.

Step 3

A total of 39 alarm floods are detected and isolated from the alarm burst rate plots similarly as shown in Figure 2.3 (assuming the number of operators is 1). The alarm flood sequences are then saved in the discussed format for further analysis.

Step 4

The alarm flood pairs which do have enough alarms in common (with Jaccard distances greater than 0.7) are pre-assigned a high distance of 1. The rest of the pairs are analyzed based on distances calculated by consecutive alarm frequencies, relative occurrences, and dynamic time warping. The pairwise distances are then used to cluster different alarm floods using the AHC algorithm and different groups of alarm floods are identified.

5.3 Results

The AHC algorithm rearranges the order of the items based on their pairwise distances. The algorithm re-orders the rows and the columns of the distance matrix in a way so that the similar items are placed together and form clusters of low pairwise distances along the diagonal. In Figure 5.2, the rearranged distance matrix for the detected 39 alarm floods is shown for distance measure $\mathcal{D}_1$ based on consecutive alarm frequency matrices. In Figures

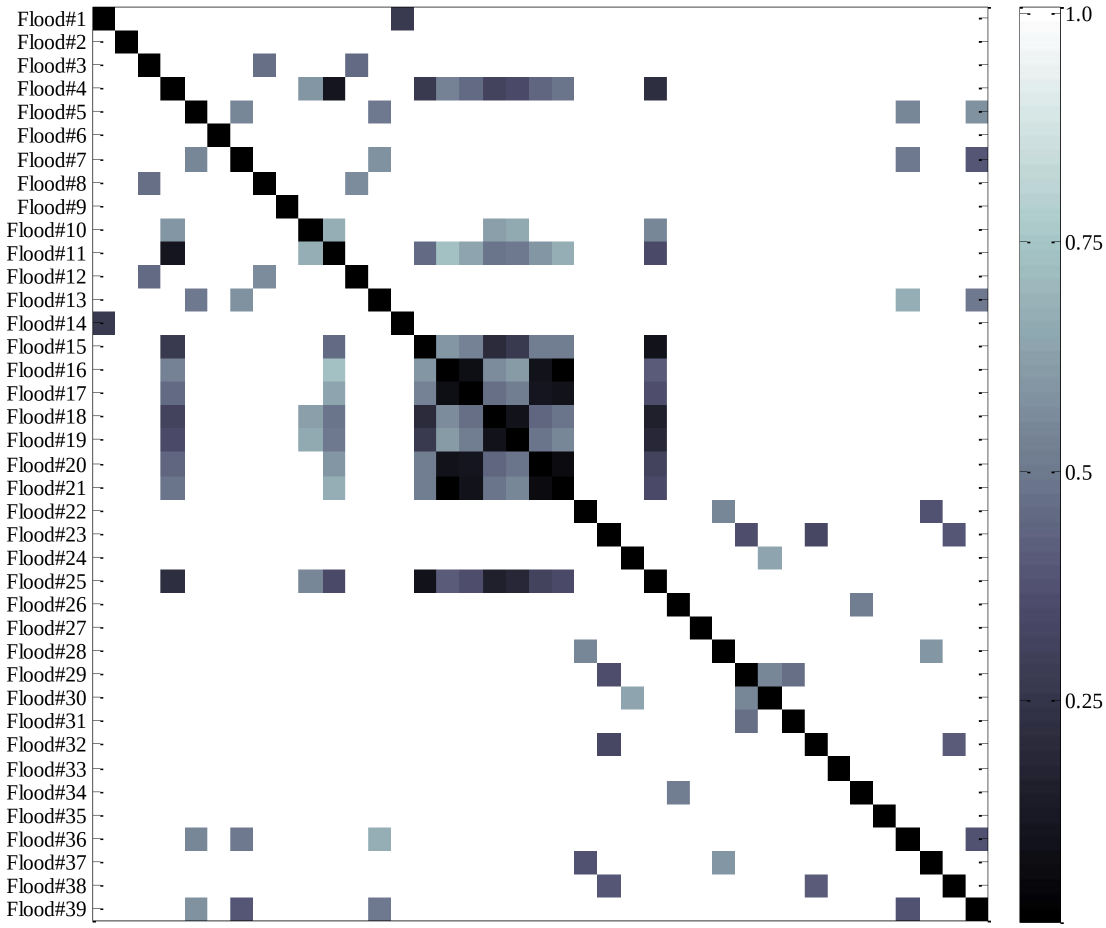


Figure 5.1: Pairwise distance matrix of 39 flood sequences. Distance measures are calculated based on consecutive alarm frequencies $\mathcal{D}_1$. The labels in the X-axis are in the same order as those in the Y-axis

5.3 and 5.4, clusters of similar floods are shown for the distances calculated based on relative occurrences of alarms ($\mathcal{D}_2$), and DTW ($\mathcal{D}_3$), respectively.

Table 5.1: Groups of alarm floods by distance measure $\mathcal{D}_1$

Groups	Flood#	Groups	Flood#
1	1,14	4	4,11,15,25,18,19
2	22,37	5	16,21,20,17
3	36,39	6	23,32

An average linkage criterion is used as the linkage function in clustering. For ease

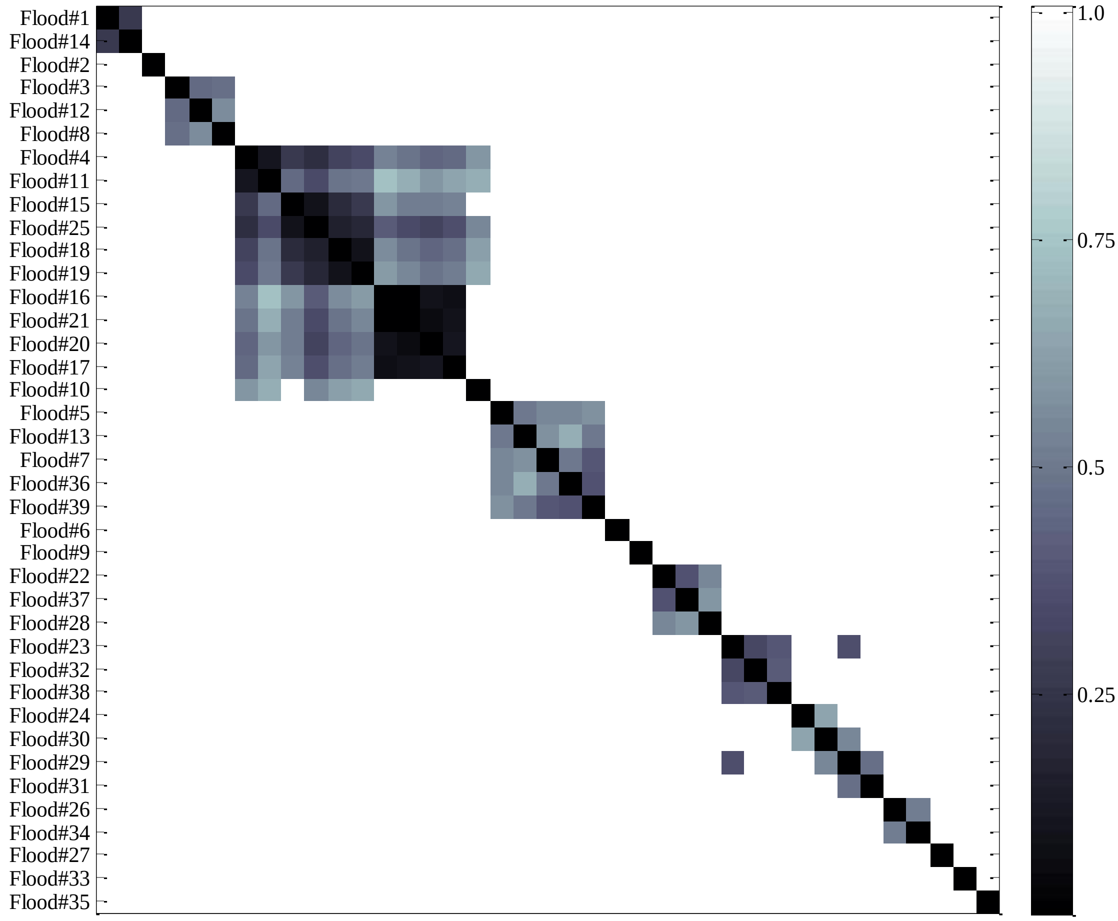


Figure 5.2: Clustered distance matrix of 39 flood sequences. Distance measures are calculated based on consecutive alarm frequencies $\mathcal{D}_1$. The labels in the X-axis are in the same order as those in the Y-axis

of visualization, the distances are color coded. The colors of the pixels in the distance matrix vary from dark to white as the similarity between alarm flood sequences changes from exactly the same to absolutely different. The dark pixels along the diagonal denote similarity with itself for an individual alarm flood. The clusters of similar alarm floods can be seen being put together in the similarity map as clusters of dark pixels. Later, according to a desired minimum similarity criterion (an upper bound on distance at 0.4), alarm floods are grouped into different classes. Different groups of alarm floods found by different distance measures are tabulated in Tables 5.1,5.2, and 5.3.

The tabulated similar alarm floods share similar patterns in alarm sequences. This similarity can be visually inspected from successful mapping between the two alarm sequences

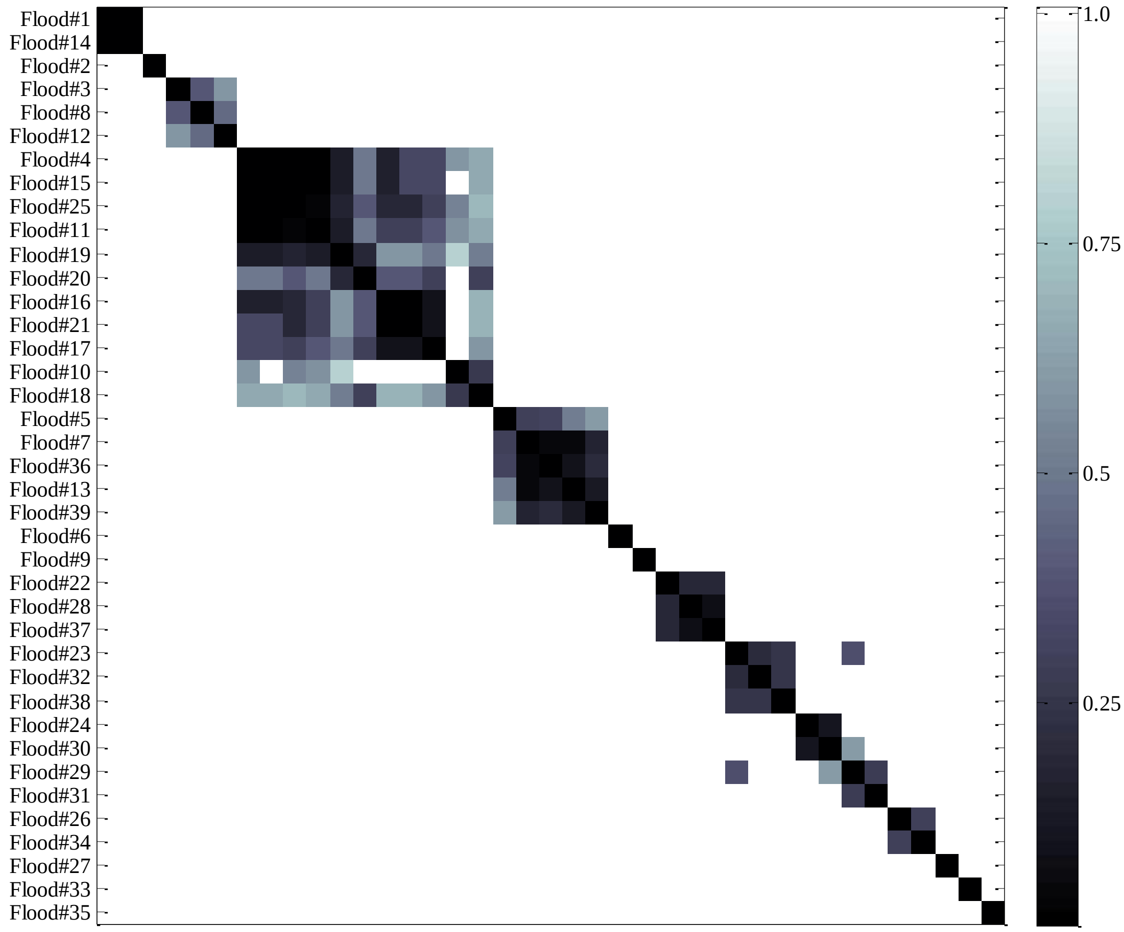


Figure 5.3: Clustered distance matrix of 39 flood sequences. Distance measures are calculated based on relative alarm occurrences $\mathcal{D}_2$. The labels in the X-axis are in the same order as those in the Y-axis

Table 5.2: Groups of alarm floods by distance measure $\mathcal{D}_2$

Groups	Flood #	Groups	Flood #
1	1,14	6	24,30
2	22,28,37	7	26,34
3	7,36,13,39	8	29,31
4	4,15,25,11,19,20,16,21,17	9	3,8
5	23,32,38	10	10,18

using DTW. A successful mapping is the one that maps a same alarm in two different sequences. In Figure 5.5, few such pairs of similar alarm floods are shown.

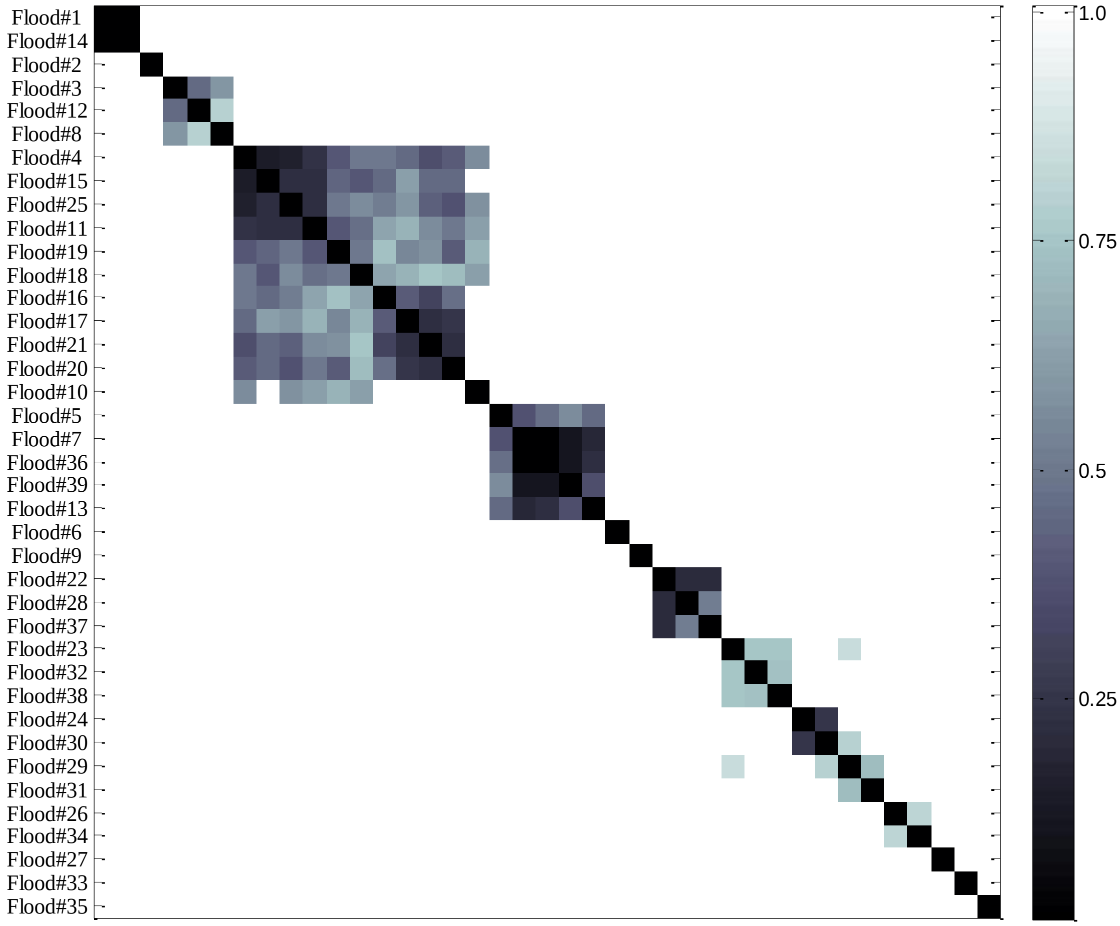


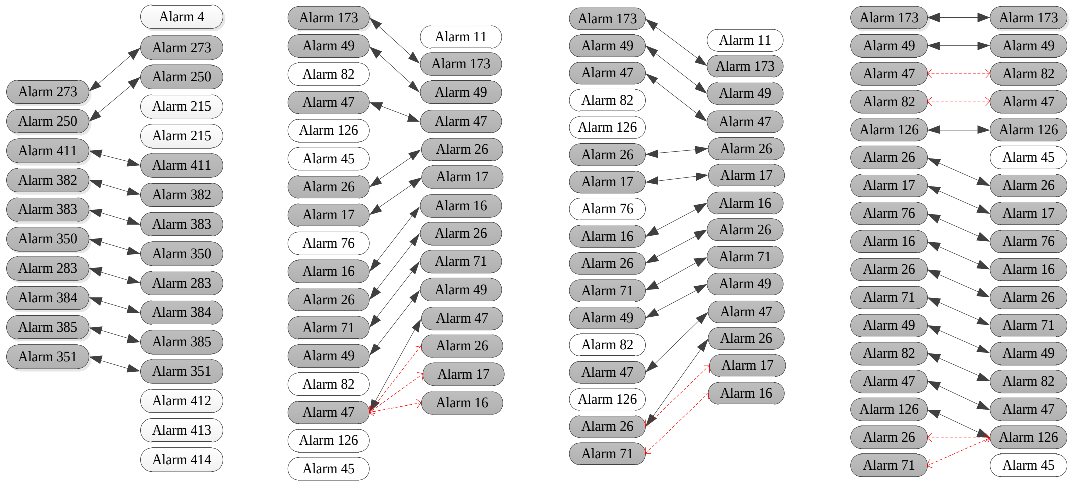
Figure 5.4: Clustered distance matrix of 39 flood sequences. Distance measures are calculated based on DTW $\mathcal{D}_3$. The labels in the X-axis are in the same order as those in the Y-axis

Table 5.3: Groups of alarm floods by distance measure $\mathcal{D}_3$

Groups	Flood #	Groups	Flood #
1	1,14	4	4,15,25,11
2	22,28,37	5	17,21,20
3	7,36,39,13	6	24,30

5.4 Discussion

From the study it can be seen that the unit has a good number of alarm floods caused by similar alarm patterns. The alarms in these patterns are very likely to be consequential



(a) flood 1 and flood 14 (b) flood 11 and flood 15 (c) flood 4 and flood 15 (d) flood 4 and flood 11

Figure 5.5: Optimal mapping between (a) flood sequence 1 and flood sequence 14, (b) flood sequence 11 and flood sequence 15, (c) flood sequence 4 and flood sequence 15, and (d) flood sequence 4 and flood sequence 11. The shaded alarms are the common alarms between two sequences in a pair. The solid mapping lines correspond to mapping of two same alarms; and dashed lines correspond to mapping of two different alarms.

rather than coincidentally following each other. Hence they are required to be studied further. With proper blending of process knowledge, many of such interrelated alarms raised in patterns could be eliminated or their root cause can be determined to rationalize the interrelated alarms. Smart alarms can also be designed to annunciate fewer alarms for such cases. Last but not the least, similar trains in future alarm flood sequences can be detected. A predetermined course of action for such alarm floods can be suggested to operators for faster and more reliable responses.

Chapter 6

Conclusions

6.1 Contribution of the Thesis

Alarm floods in process industry is a common and serious problem. The complexities involved in the study of multiple process variables in large systems pose obstacles to perform critical analysis. As a result, alarm flooding has been becoming more and more serious in the process industry. In this work, alarm floods have been discussed considering the standards provided in ISA 18.2 and the guidelines in EEMUA 191. A method to represent alarm data which can effectively capture alarm activities and formation of alarm floods has been presented. The effect of chattering alarms on alarm floods has been discussed and it has been shown how chattering alarms can be removed for a better alarm flood analysis. A new analysis on alarm floods has been proposed to classify recorded alarm floods based on the similarity of alarm sequences. Three different similarity measures have been discussed which can capture patterns in alarm annunciations. A case study on real industrial data has been shown to demonstrate the utility of the proposed analysis.

Similarity analysis or pattern classification of alarm floods is a new area of research in alarm management. In an alarm management life cycle, critical analysis of alarm systems with respect to a alarm flood problems is essential for routine updates and improvements. The similarity analysis of alarm floods presented in this thesis is independent of process knowledge and can be easily adopted for studying recorded alarm floods in the process industry. Similar sequences in alarm annunciation during different alarm floods are most likely to be the results of undiscovered interrelations among process variables. The results of the proposed analysis can be used to identify such possible interactions and eliminate sequential alarms. It is also possible to ascertain rationalization suggestions for interrelated alarms, which are the main objective in a monitoring and assessment stage in an alarm management life cycle.

6.2 Scope for Future Work

In this work, only timestamped alarm messages have been used to cluster similar alarm floods. The similarity investigation can be further improved by combining alarm information and operator's actions together. In modern industries, every action and process variable is recorded in the master database. These information are usually not used in a proper manner because of complexities involved in managing large data. Intelligent analysis of industrial data by linking information from different sectors of a process could reveal many interesting relationships. It is also interesting to expand this work to include causality analysis of the process variables responsible for similar alarm sequences. The result of such an analysis will help in designing smart and complex alarms associated with multiple related variables in industry.

Bibliography

[1] I. Izadi, S. L. Shah, D. S. Shook, and T. Chen, "An introduction to alarm analysis and design," in *Proceedings of the 7th IFAC SAFEPROCESS*, pp. 645–650, 2009.

[2] P. D. Christofides, J. F. Davis, N. H. El-Farra, D. Clark, K. R. D. Harris, and J. N. Gipson, "Smart plant operations: vision, progress and challenges," *AIChE*, vol. 53, no. 11, pp. 2734–2741, 2007.

[3] "Abnormal situations management consortium." http://www.asmconsortium.com.

[4] M. Bransby and J. Jenkinson, *The management of alarm systems.* Sudbury: HSE Books, 1998.

[5] I. Izadi, S. L. Shah, D. S. Shook, S. R. Kondaveeti, and T. Chen, "A framework for optimal design of alarm systems," in *Proceedings of the 7th IFAC SAFEPROCESS*, pp. 651–656, 2009.

[6] The International Society of Automation (ISA), *ANSI/ISA-18.2-2009, Management of alarm systems for the process industries*, 2009.

[7] Engineering Equipment and Materials Users Association (EEMUA), *Alarm systems: a guide to design, management and procurement*, 2007.

[8] H. G. Kang and P. H. Seong, "A methodology for evaluating alarm-processing systems using informational entropy-based measure and the analytic hierarchy process," *IEEE Transactions on Nuclear Science*, vol. 46, no. 6, pp. 2269–2280, 1999.

[9] K. Julisch, "Mining alarm clusters to improve alarm handling efficiency," in *Proceedings 17th Annual Computer Security Applications Conference (ACSAC)*, pp. 12–21, 2001.

[10] G. Lambert-Torres, E. F. Fonseca, M. P. Coutinho, and R. Rossi, "Intelligent alarm processing," in *International Conference on Power System Technology*, pp. 1–6, 2006.

[11] J. W. Stahlhut, G. T. Heydt, and J. B. Cardell, "Power system 'economic alarms'," *IEEE Transactions on Power Systems*, vol. 23, no. 2, pp. 426–433, 2008.

[12] L. Yan, L. Xiwei, M. Noda, and H. Nishitani, "Systematic design approach for plant alarm systems," *Journal of Chemical Engineering of Japan*, vol. 40, no. 9, pp. 765–772, 2007.

[13] J. Ahnlund and T. Bergquist, "An alarm reduction application at a district heating plant," in *Proceedings of IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 2, pp. 187–190, 2003.

[14] T. Bergquist, J. Ahnlund, and J. Larsson, "Alarm reduction in industrial process control," in *Proceedings of IEEE Conference on Emerging Technologies and Factory Automation (ETFA), 2003.*, vol. 2, pp. 58–65, 2003.

[15] Y. Cheng, I. Izadi, and T. Chen, "On optimal alarm filter design," in *International Symposium on Advanced Control of Industrial Processes (ADCONIP)*, pp. 139 –145, 2011.

[16] N. A. Adnan, I. Izadi, and T. Chen, "Computing detection delays in industrial alarm systems," in *Proceedings of the American Control Conference*, pp. 786–791, 2011.

[17] E. Naghoosi, I. Izadi, and T. Chen, "A study on the relation between alarm deadbands and optimal alarm limits," in *Proceedings of the American Control Conference*, pp. 3627–3632, 2011.

[18] S. R. Kondaveeti, I. Izadi, D. S. Shah, S. L.and Shook, and R. Kadali, "Quantification of alarm chatter based on run length distributions," in *49th IEEE Conference on Decision and Control (CDC)*, pp. 6809 –6814, 2010.

[19] S. R. Kondaveeti, I. Izadi, S. L. Shah, and T. Black, "Graphical representation of industrial alarm data," in *Proceedings of the 11th IFAC Symposium on Analysis, Design and Evaluation of Human-Machine Systems*, 2010.

[20] J. Nishiguchi and T. Takai, "Ipl2 and 3 performance improvement method for process safety using event correlation analysis," *Computers & Chemical Engineering*, vol. 34, no. 12, pp. 2007 – 2013, 2010.

[21] S. R. Kondaveeti, S. L. Shah, and I. Izadi, "Application of multivariate statistics for efficient alarm generation," in *Proceedings of the 7th IFAC SAFEPROCESS*, pp. 657–662, 2009.

[22] I. Izadi, S. Shah, and T. Chen, "Effective resource utilization for alarm management," in *49th IEEE Conference on Decision and Control (CDC)*, pp. 6803 –6808, 2010.

[23] F. Yang, S. Shah, D. Xiao, and T. Chen, "Improved correlation analysis and visualization for industrial alarm data," in *Accepted for 18th IFAC World Congress*, 2011.

[24] K. Hlavckov-Schindler, M. Palus, M. Vejmelka, and J. Bhattacharya, "Causality detection based on information-theoretic approaches in time series analysis," *Physics Reports*, vol. 441, no. 1, pp. 1 – 46, 2007.

[25] L. Barnett, A. B. Barrett, and A. K. Seth, "Granger causality and transfer entropy are equivalent for gaussian variables," *Physical Review Letters*, vol. 103, no. 23, p. 238701, 2009.

[26] M. Bauer, J. W. Cox, M. H. Caveness, J. J. Downs, , and N. F. Thornhill, "Finding the direction of disturbance propagation in a chemical process using transfer entropy," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 1, pp. 12 –21, 2007.

[27] F. Yang, D. Xiao, and S. L. Shah, "Optimal sensor location design for reliable fault detection in presence of false alarms," *Sensors*, vol. 9, no. 11, pp. 8579–8592, 2009.

[28] F. Yang, S. L. Shah, and D. Xiao, "SDG (signed directed graph) based process description and fault propagation analysis for a tailings pumping process," in *Proceedings of the 13th IFAC Symposium on Automation in Mining, Mineral and Metal Processing*, pp. 50–55, 2010.

[29] F. Yang, S. L. Shah, and D. Xiao, "Signed directed graph modeling of industrial processes and their validation by data-based methods," in *Proceeding of the 2010 Conference on Control and Fault-Tolerant Systems (SysTol)*, pp. 387 –392, 2010.

[30] X. Liu, M. Noda, and H. Nishitani, "Evaluation of plant alarm systems by behavior simulation using a virtual subject," *Computers & Chemical Engineering*, vol. 34, no. 3, pp. 374 – 386, 2010.

[31] A. L. Delcher, D. Harmon, S. Kasif, O. White, and S. L. Salzberg, "Improved microbial gene identification with glimmer," *Nucleic Acids Research*, vol. 27, no. 23, pp. 4636–4641, 1999.

[32] A. V. Lukashin and M. Borodovsky, "Genemark.hmm: new solutions for gene finding," *Nucleic Acids Research*, vol. 26, no. 4, pp. 1107–1115, 1998.

[33] J. Henderson, S. Salzberg, and K. H. Fasman, "Finding genes in DNA with a hidden markov model," *Journal of Computational Biology*, vol. 4, pp. 127–141, 1997.

[34] S. L. Salzberg, A. L. Delcher, S. Kasif, and O. White, "Microbial gene identification using interpolated markov models," *Nucleic Acids Research*, vol. 26, pp. 544–548, 1998.

[35] H. Wang, C. S. Perng, W. Fan, S. Park, and P. S. Yu, "Indexing weighted-sequences in large databases," in *Proceedings of 19th International Conference on Data Engineering, 2003*, pp. 63–74, 2003.

[36] S. Park, J. I. Won, J. H. Yoon, and S. W. Kim, "A multi-dimensional indexing approach for timestamped event sequence matching," *Information Sciences*, vol. 177, no. 22, pp. 4859–4876, 2007.

[37] B. R. Hollifield and E. Habibi, *Alarm Management - Seven Effective Methods for Optimum Performance*, ch. 7, pp. 44–47. ISA, 2007.

[38] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[39] R. O. Duda and P. E. Hart, *Pattern classification and scene analysis.* New York: Wiley-Interscience Publication, 1973.

[40] C. D. Manning, P. Raghavan, and H. Schütze, *An Introduction to Information Retrieval*, ch. 17, pp. 377–401. New York, NY, USA: Cambridge University Press, 2009.

[41] M. J. Lesot, M. Rifqi, and H. Benhadda, "Similarity measures for binary and numerical data: a survey," *International Journal of Knowledge Engineering and Soft Data Paradigms*, vol. 1, no. 1, pp. 63–84, 2009.

[42] R. Bellman, *Dynamic Programming.* Dover Publications, 2003.

[43] L. Rabiner and B.-H. Juang, *Fundamentals of speech recognition.* NJ, USA: Prentice-Hall, Inc., 1993.

[44] M. Müller, *Information Retrieval for Music and Motion.* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[45] S. Park, W. Chu, J. Yoon, and C. Hsu, "Efficient searches for similar subsequences of different lengths in sequence databases," in *Proceedings of 16th International Conference on Data Engineering*, pp. 23 –32, 2000.