A Framework for discovering Interesting Rules from Event Sequences with the purpose of pre-warning Oil Production Problems

Joacim Lunewski Christiansen

Master of Science in Computer Science
Submission date: June 2007
Supervisor: Torulf Mollestad, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Problem Description

ConocoPhillips wants to explore the problem of how to reveal future problems in a production process, produce appropriate alarms and thereby allowing efficient handling of an unwanted and potentially dangerous situation. The problem is to investigate how rules suited for prediction of future production problems can be found from a sequence of events gathered from a production system. This thesis is part of a bigger process where the idea is to go from raw data to rules that can be combined in a reasoning structure for predicting future problems in a production process.

Assignment given: 16. January 2007
Supervisor: Torulf Mollestad, IDI

Abstract

Periods of sub-optimal production rates, or complete shut-downs, add negative numbers to the revenue graph for oil companies. Oil and gas are produced from several reservoirs and through many wells with varying gas/oil proportion, making it a complex process that is difficult to control. As a part of a three step process for utilizing data in the oil production domain, this thesis derive methods for discovering event patterns, called restricted association rules, from time series in order to pre-warn about future problems in oil production processes. A restricted rule syntax and semantics is derived to explicitly target rules suited for prediction. Based on the defined rule syntax, a two step process is derived where restricted rule mining based on the concept of minimal occurrences is used to discover restricted association rules from a sequence of events. Next, redundant rules are removed based on the concept of minimum improvement and chaining of rules, during a rule selection phase. Information theory is applied in order to identify the most interesting rules, which can be submitted to an expert for validation. Both a simple solution for easy implementation in ConocoPhillips and a more advanced solution appropriate for general prediction cases are derived. This thesis concludes that it is feasible to discover dependencies between events from actual process data. It is also concluded that a large number of rules can be pruned, in order to get a manageable set of rules which is believed to have good predictive performance.

Preface

"The golden rule is that there are no golden rules."
– George Bernard Shaw (1856 - 1950)

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) as a part of my Master in Computer Science. The work contained herein has been performed at the Department of Computer and Information Science, NTNU, Trondheim, under the supervision of Associate Professor II Torulf Mollestad.

This thesis has been written as my final work as a fifth year student. My background is mainly in computer science with a specialization in knowledge systems, and an interest for complex algorithms. Taking on the problems of oil production has been a challenge, but at the same time inspiring. While writing this thesis I have encountered numerous exiting problems and been introduced to a variety of new fields of research. It has been a though way, but at the same time great fun. Looking back, I realize that this basically is the story of these five years at NTNU; great challenges combined with inspiring and great people.

During the last year, and many late nights, nothing has been more fruitful than discussions with my trusted companion Per Kristian Helland. I want to thank him for good discussions and the guts to go on even further. This thesis had not been the same without this inspiring teamwork.

My supervisor, Torulf Mollestad, has made his expertise available and been very helpful. Great freedom has been given, but combined with clear and informative feedback I have been forced to rethink my beliefs and think out of the box. Thank you Torulf, for the invaluable support and advice.

Trondheim, 07/06/2007

Joacim Christiansen

Contents

List of Tables

List of Figures

Chapter 1

Introduction

For oil companies, the combination of today's record-high oil prices and stakeholders strict demands for return on investments, combined with sparse oil and gas resources, brings forth the need for business optimization. Onshore operation centres (OOC) are extensively supporting offshore operations using state of the art communication technology. This reduces the need for offshore personnel and is a cost effective way of managing operations. It also introduces the possibility for additional monitoring systems that can be operated onshore.

Oil production is a non-trivial process and it differs in nature from reservoir to reservoir. More sophisticated technology makes it possible to do drilling operations that would have been impossible only few years ago, like horizontal drilling. The new possibilities also increase the complexity of oil production since e.g. oil and gas are produced from several reservoirs and through many wells with varying gas/oil proportion.

In oil production, periods of unplanned sub-optimal production rates, or complete shut-downs, add negative numbers to the revenue graph. For instance, ConocoPhillips loses 1.4 million Danish Kroner (about 175 000 Euro) in income for each hour an oil well is not in production [14]. On average, ConocoPhillips calculates with around 40 unplanned well shut downs per year, and the total costs become severe. A system that could give an early warning about possible future anomalies in the oil production, and do this in time for the operators to react, would therefore be valuable in order to reduce costs.

1.1 Research goals

Monitoring of oil production processes in ConocoPhillips is used in order to ensure that the process is effective. Such monitoring usually involves several sensors, which are used to depict a variety of attributes of the process over time. Examples of attributes are temperature, pressure and valve openings, among others. Measurements are done on those parts of the system which are considered critical for the operation and process stability. The choice of attributes to monitor is critical, and is mainly performed by human experts. The frequency of sampling is chosen by the user and may vary from a few milliseconds to many days dependent on the attribute in question, and may also differ over time. The observation data, and measurement of production rates, are stored in time series databases.

For ConocoPhillips, elimination of production loss is an important goal, and this includes prediction of possibly production stops and loss of quality in production. Both complete halt in oil production and sub-optimal production rates result in a considerable loss of money. An example of a possible production rate time series is given in Figure 1.1. It illustrates a characteristic full stop in production ,and an interval of sub-optimal production (shaded area).

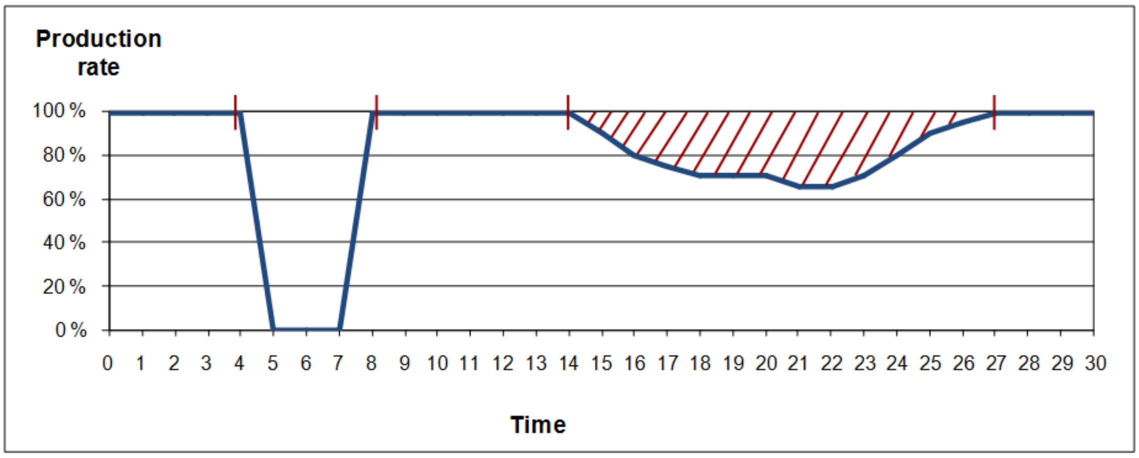


Figure 1.1: Example illustrating sub-optimal production

In general, methods for searching large volumes of data for patterns belong to the field of data mining. One goal is to characterise the general properties of a data collection and/or perform inference on the current data in order to make predictions. The monitoring of sensors in oil production produces a large amount of data, providing a breeding ground for a data mining hypothesis that patterns can be found in the historical data, and subsequently be applied to real-time production data in order to warn about future production rate anomalies.

1.1.1 Problem description

ConocoPhillips wants to explore the problem of how to reveal future problems in a production process, produce appropriate alarms and thereby allowing efficient handling of an unwanted and potentially dangerous situation. The problem is to investigate how rules suited for prediction of future production problems can be found from a sequence of events gathered from a production system. This thesis is part of a bigger process where the idea is to go from raw data to rules that can be combined in a reasoning structure for predicting future problems in a production process.

The problem of finding event based rules suited for prediction is divided into three sub-problems:

1. **Characteristics of rules suited for prediction:** Defining the characteristics of rules suited for prediction of time based events.

2. **Rule mining from event sequences:** How to apply an association rule mining approach to event sequences.

3. **Finding interesting rules:** How to identify those rules which are truly interesting from a large rule set, in the context of predicting production problems.

In a broader perspective this thesis contributes to the field of rule mining and interestingness of rules, with a specification towards event based prediction. The thesis should mainly have a theoretical approach to the problem, but with a pragmatic view.

1.1.2 Limitations of scope

This master thesis' main goal is to reach a conclusion of the feasibility of discovering association rules from event sequences in the oil production domain. It will not consider the challenge of integrating suggested solutions with existing systems in ConocoPhillips.

It is also important to note the difference between predicting the production rate and predicting possible production anomalies. Predicting production rate may be seen as predicting a graph based only on the history of this single graph, analogue to predicting a stock price. Predicting possible production anomalies is a broader problem concerning prediction of variables contributing to a higher level production anomaly. This work contributes to the latter, with the ultimate goal of giving early warnings of faults in order to make them avoidable.

1.2 Background and motivation

ConocoPhillips was founded in 2002 after the merging of Conoco Inc. and Phillips Petroleum Company. It is among the largest non-government controlled energy companies worldwide, having, at year-end 2005, 38 400 employees and assets of $165 billion USD [5]. ConocoPhillips' core activities worldwide are:

- Petroleum exploration and production.
- Petroleum refining, marketing, supply and transportation.
- Natural gas gathering, processing and marketing.
- Chemicals and plastics production and distribution.

The oil industry is one of the largest and most profitable industries in the world. It is also one of the most cost aware industries. Since the price of oil is standardized, oil companies concentrate on cutting internal production costs and optimizing production.

ConocoPhillips Norge accounts for about 14 percent of the oil and gas production in ConocoPhillips worldwide. It has a total of 1735 employees and is the third-largest energy company in Norway [14]. In January 2005, ConocoPhillips started a program called From Good to Great which directly involves over 700 employees. The goal of From Good to Great is to improve processes and increase efficiencies to cut costs and optimize production. The hierarchy of these goals are:

1. Maximize production and increase the process regularity.
2. Reduce costs.

Fault detection is a key issue in the development of complex oil production facilities. Several techniques, including alarms on sensors and process simulation have been tried in order to avoid failures or loss in production, with varying degree of success. Monitoring based on process knowledge is mainly done by domain experts, but as the complexity of these systems grows this becomes a difficult task to manage. Such monitoring requires a tremendous amount of a priori knowledge of the system behaviour for each facility, whereas a large part of this knowledge is often tacit and not easily obtained. For example, deriving a complete set of production rules for a facility is usually a time consuming task, and too expensive to do for each facility. In addition, re-use of knowledge from other facilities is limited, because each facility is usually more or less different from others. Another aspect is introducing new operators, which in general is a huge challenge because of the tacit and not easily obtained knowledge. Therefore another goal is to make this knowledge easily available to new operators.

Prediction problems have received great attention in the statistical and financial domain, e.g. attempts to predict stock prices, but results have been relatively bad with respect to correctness of predictions. While the stock market is a non-deterministic process with a huge amount of uncertainty, the oil production process can be assumed to be deterministic and describable by a finite set of states. Oil production is limited by physical laws and tends to operate in a closed environment. This and the hypotheses that there exists a logical and statistical correlation between sensor data and production anomalies that may be found, motivates for a possible data mining solution.

1.3 Data to Decision: The greater picture

One pillar of From Good to Great is Data to Decision (D2D). D2D directly involves around 40 employees and is the IT infrastructure to support From Good to Great. It is applicable in one oil field in the North Sea named Ekofisk. This field represents approximately 10% of the oil company's worldwide oil production. The daily field production is an estimated 375 000 barrels of oil, equaling around 22 million USD in daily revenues [14]. D2D deals with the problem of how the vast number of stored data related to oil production can be used to make faster and better decisions.

1.3.1 Remote decision making

An ultimate goal for data utilization, in the context of predicting production anomalies, is to create a failure prediction system. A likely scenario for a system of that kind is that it will be integrated in an OOC. An OOC is a digital operation centre located onshore, which is connected to offshore drilling and production facilities by fibre wires. Figure 1.2 places OOCs in the greater picture of what is called Integrated Operations (IO), the future of oil production, where technology that gathers competence, data and applications in real-time, independent of distance, is extensively utilized. Information sent from operational instruments makes it possible to monitor and control the platforms remotely. Daily meetings between the operational management located onshore and offshore have brought the two environments closer. In addition, real-time data is distributed to collaborators, like maintenance actors. A focus area for the OOC of ConocoPhillips Norway is production optimization, and elimination of production loss is an important goal.

The technology available at an OOC makes it possible to operate a failure prediction system there, and the vision is to build a system that receives real-time sensor data from oil production as input, and con-

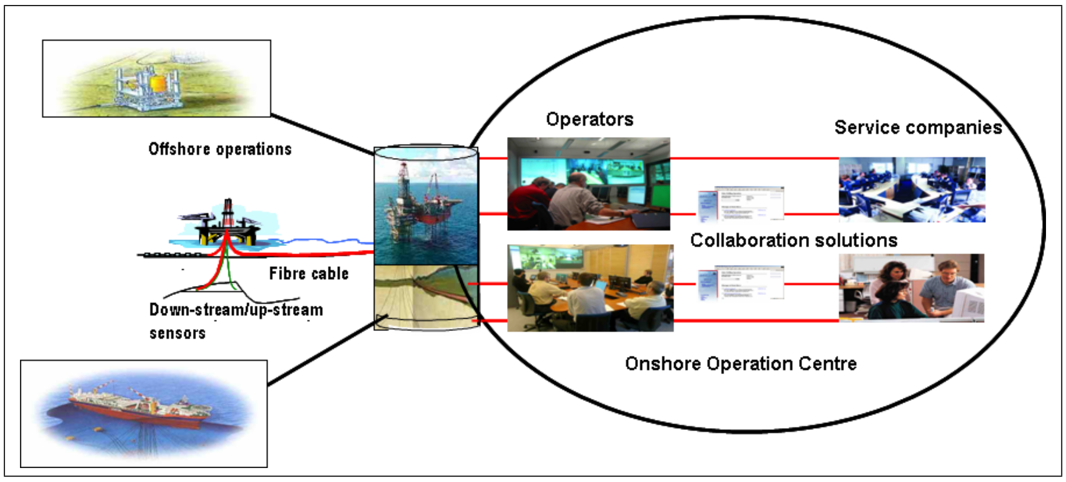


Figure 1.2: Integrated operations

tinuously processes the data in order to discover important events and reason about possible failures. Ideally the production state should be visualised on a large screen, showing events and their dependencies. If a warning is given, the system should explain the cause and suggest corrective actions. Such a system will probably reduce the amount of tacit knowledge needed when monitoring oil production because this knowledge is communicated through the visualised production state. The ability to predict more complex situations early will probably be increased since the prediction, which might include a great amount of data, is handled by computers.

1.3.2 Problem case

The specific production process that has been used as a test case in D2D is named "2/4 J" and is sketched in Figure 1.3.

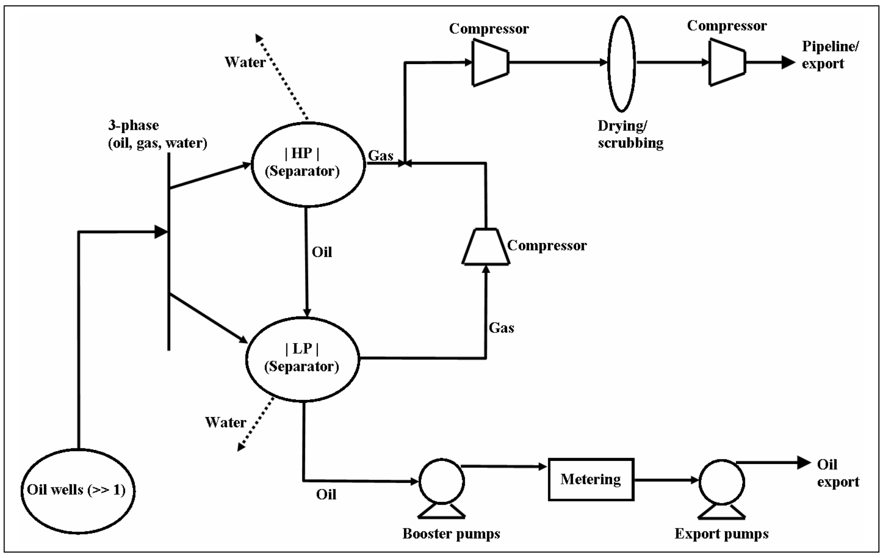


Figure 1.3: Production process 2/4 J

This simplified figure shows how 3-phase flows of gas, oil and water are lead into a high pressure or a low pressure separator. The water is cleaned and then discharged into the sea, while oil and gas is sent to onshore refineries. Throughout this process sensors[1] are monitoring various attributes, as mentioned earlier.

An emulsion is a mixture of two immiscible substances, and for a 3-phase separator this can happen for oil/water. The problem case targeted for prediction is emulsification in the low pressure separator of the 2/4 J process. A sketch of the separator is shown in Figure 1.4.

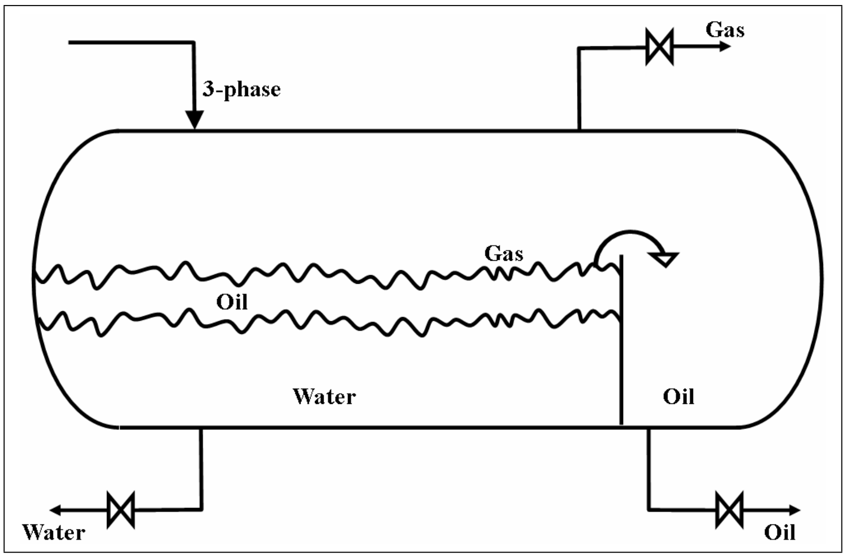


Figure 1.4: A simplified low pressure separator

As shown, oil and water leave the vessel at the bottom through different valves, and the gas leaves the vessel at the top. If the water level is to high, both water and oil will flow over the edge. A high amount of water gives low quality oil, and hence a lower price. If emulsification occurs, the water level is impossible to measure, and corrective actions have to be done.

1.3.3 Guidelines for data utilization

This study is related to D2D in general, and in particular the work done by Senior Consultant at SAS Institute, and Associate Professor II at NTNU, Torulf Mollestad. Mollestad [32, 33] has developed a general guideline for data utilization in the oil production domain, and under his supervision a project thesis investigating this guideline was written by Christiansen and Helland [11] autumn 2006. Figure 1.5 shows the suggested three-step process, going from sensor data to an operational system.

The process illustrated by Figure 1.5 is not trivial, and each step is explained next, in more detail.

1. Data preprocessing includes data loading, data cleaning, event discovery and variable clustering, as explained in Table 1.1.

For the particular problem case of emulsification in the low pressure separator of process 2/4 J, Mollestad [32, 33] has developed a framework for finding relevant properties of sensor data. The framework can be

[1]The terms "sensor" and "tag" are used alternating.

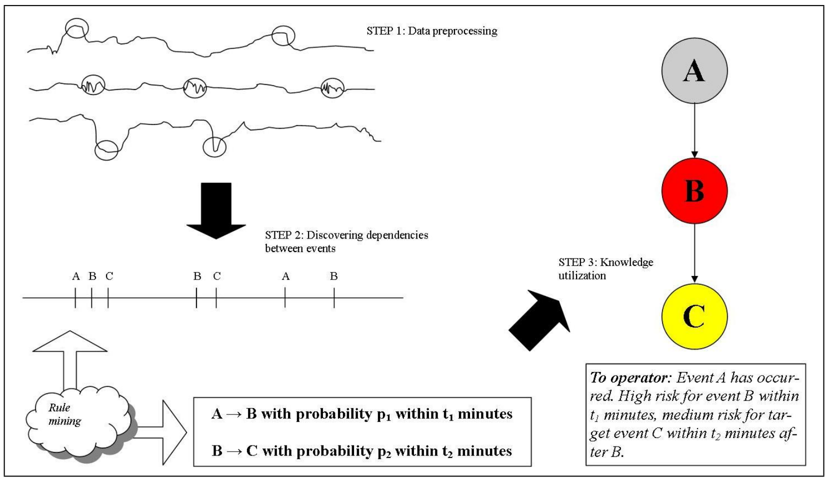

STEP 1: Data preprocessing
STEP 2: Discovering dependencies between events
STEP 3: Knowledge utilization
A B C B C A B
A
B
C
Rule mining
A → B with probability p_1 within t_1 minutes
B → C with probability p_2 within t_2 minutes
To operator: Event A has occurred. High risk for event B within t_1 minutes, medium risk for target event C within t_2 minutes after B.

Figure 1.5: Illustration of the overall three-step data mining vision.

divided into two main steps:

- *Data transformation:* Let domain experts select tags (sensors) that are considered potentially relevant. Apply transformation functions to each time series (data from selected tags) in order to derive binary event time series. Examples of transformation functions are: Extremely high/low levels, high volatility in data, long/dramatic shifts in the data, passing of (predefined) critical levels, and user defined functions. In order to continue to the next step, a target event must be defined by domain experts.
- *Variable clustering:* Cluster the binary event time series in order to identify events that co-occur with the target event. The goal of this process is to remove uninteresting and/or non-significant variables, and the resulting events are those related to problem situations, i.e. they are potential warning signals.

For the 2/4 J process, 119 tags were selected as potentially relevant, and the transformation process generated 1050 event time series. The target event defined by experts was identified to happen 22 times in the data from December 2005 to December 2006. The variable clustering then reduced the set of binary event time series to about 200.

2. Discovering dependencies between events (rule mining) includes the concepts of sliding window extraction, event sequence mining, restricted association rules, interestingness measures and rule selection, as explained in Table 1.2.

3. Knowledge utilization is the final step and involves learning of a network structure, definition of network reasoning and operational use. The concepts are further explained in Table 1.3.

Parallel to this thesis work on the third step has been done by Helland [20]. Hellands work is done in close cooperation with the work done in this thesis because it relies on some of the findings from this thesis. The main findings of Helland's work are two-folded. First, an algorithm is suggested for learning a generic network structure from a set of rules. The network is built as a hypergraph, meaning that edges can span several nodes, and is free of cycles. Nodes and edges in the network corresponds to events and rules, respectively. Second, a transformation of the network to a DAG is described, making it possible to use simple reasoning mechanisms for explaining correlations between nodes. A more advanced solution, based on the concept of Bayesian networks, is also given. In general, the work contributes to the field of expert systems, combining what is called local and global methods of data mining.

This thesis is placed between the work done by Mollestad and Helland, and concerns the second step in the process. It is grounded on Mollestad's work in the sense that it is assumed that the problem of discovering events in ConocoPhillips is solved, and that the events found are somewhat relevant for prediction. It should be noted that the output from the work done in this thesis is the natural input for Helland's work, and that the goal of predicting failures can not be solved by the work done in this thesis alone. Hence, will this work be constrained to an output that makes reasoning possible.

Task	**Description**
Data loading	The process of loading sensor data from the database where it is located. The amount of available data is in many cases greater than what is manageable for further processing. A decision of what sampling rate to use must be taken.
Data cleaning	The process of filling in missing values, smoothing out noise and correction of data inconsistency.
Event detection	The process of discovering unusual and/or important events in the time series. An event must be explainable to a user in order to be useful. Basic techniques as threshold measures are often suitable because they are computational efficient and easy to interpret.
Variable clustering	The process of removing variables (event series) that do not influence on production loss. Removing such uncorrelated variables, which also may be seen as noise, is beneficial because it may reduce running times and improve the results of association rule mining algorithms.

Table 1.1: Step 1: Data preprocessing

Task	**Description**
Sliding window extraction	The process of splitting a sequence of different events into groups of events occurring within some fixed time interval.
Extended association rules	Association rules which describes association between occurrences of events or combination of events, taking temporal relations into account. Such rules may be found in the historical data and used to predict future events.
Interestingness measures	Primarily mathematical expressions that represent the interestingness of rules. Such expressions may be used to efficiently prune the search space during rule mining, but also in rule cleaning/selection.
Rule selection	The process of identifying interesting rules and removing non-interesting, redundant or unwanted rules. Such a process may be done automatically or by a human expert using an interactive rule selection tool.

Table 1.2: Step 2: Discovering dependencies between events

Task	**Description**
Network learning	The task of building a network based on the discovered extended association rules and their dependencies.
Network reasoning	The process of drawing inferences appropriate to the situation, i.e. update the belief on events in order to predict failures.
Operational use	Consists of to parts: (1) How to make suggestions of corrective actions (technical), and (2) how to integrate the system with other monitoring systems (system integration).

Table 1.3: Step 3: Knowledge utilization

1.4 Report outline

The key issue in this thesis is how to discover association rules from event sequences that are suited for prediction of production failures. This problem is divided into three logical sub-problems: 1) Is it possible to define some properties of rules that have a single purpose of prediction, which can be used to efficiently discover only such rules? 2) How can such rules be discovered from event sequences? 3) How can uninteresting rules, with respect to prediction, be pruned from a large rule set, giving only rules interesting for prediction?

The answer to these questions are given in the three main chapters of this report:

1. Rule syntax and semantics
2. Rule mining
3. Rule selection

First, in Chapter 2 the required theoretical background needed for understanding the technical framework developed in this thesis will be explained. Chapter 3 defines a rule syntax and semantics that can be used to target rules suited for prediction. Chapter 4 introduce how rules can be discovered from event sequences and derives an approach for restricting the rule mining to rules suited for prediction. Chapter 5 evaluates different measures of interest for rules, and suggests a rule selection approach for removing rules not interesting for prediction. In Chapter 6 the various issues encountered during this thesis is discussed. Finally, in Chapter 7 a conclusion on the feasibility of the derived approach is done before an overview of possible extensions of this work is given.

Chapter 2

Preliminaries

This chapter will introduce the required theoretical background needed for understanding the technical aspects in this thesis; association rules and information theory. The thesis is founded on the existing framework of association rules and association rule mining. Understanding the association rule paradigm thus becomes important for understanding the theoretical aspects of this thesis. Information theory provides a formalism for expressing the amount of information described or contained in rules. A measure of information has proven to be a good differentiator between rules used for classification and prediction. These two theoretical foundations are described in the following sub-chapters.

2.1 Association rule mining

Association rule mining is one of the dominant techniques in the data mining literature, and has received much interest in the research community. The task of association rule mining was introduced in 1993 [1] and has since been used in many application domains. Looking back, the motivation for mining association rules was the great amount of *basket data*; items purchased on a per-transaction basis in super markets, stored in transaction databases. Finding association and buying patterns between products can be an important knowledge when it comes to decision making and marketing.

Adopting the formal definition given by Hipp et al. [23], let $\mathcal{I} = \{x_1, ..., x_n\}$ be a set of distinct literals, called items (e.g. articles or products). Also, let the database $\mathcal{D}$ be a multi-set of subsets of $\mathcal{I}$ where each $T \in \mathcal{D}$ is called a transaction (e.g. if T consists of articles bought by one particular customer, then $\mathcal{D}$ consists of all such purchases). A set of items, $X \subseteq \mathcal{I}$, is referred to as an *itemset*, or more often a k-itemset where k = $|X|$. An association rule is on the form $X \to Y$ where X and Y are itemsets and $X \cap Y = \emptyset$ (following the example so far, X may be bread and Y milk). Two important measures are defined from this:

1. **Support:** A transaction $T \in \mathcal{D}$ supports an itemset $X \subseteq \mathcal{I}$ if $X \subseteq T$ holds. The support of the rule $X \to Y$ is the percentage of transactions in $\mathcal{D}$ that contain X and Y. This is defined as

$$supp(X \to Y) = \frac{\#(X \text{ and } Y)}{\#(T \in \mathcal{D})} \tag{2.1}$$

2. **Confidence:** The confidence of a rule $X \rightarrow Y$ is the percentage of transaction in $\mathcal{D}$ containing X that also contain Y. This is defined as

$$conf(X \rightarrow Y) = \frac{supp(X \rightarrow Y)}{supp(X)} \tag{2.2}$$

An itemset is called *frequent*[1] if it satisfies a specified minimum support threshold. The task of association rule mining is based on finding all frequent itemsets. The dominant technique in the literature for association rule mining is the A-priori algorithm [1, 2]. In A-priori, the support measure is used as a heuristic to confine the search for frequent itemsets. The A-priori algorithm takes advantage of the fact that all non-empty subsets of a frequent itemset must also be frequent. By joining frequent (k-1)-itemsets to generate candidate k-itemsets, pruning the candidate set can be done by deleting those containing any subset that is not frequent. Then the candidate itemsets are checked against the data, and the support of the itemsets computed. Those itemsets proven to be frequent are the basis for the next candidate generation. This process is repeated until all possible frequent itemsets are found. From the set of all frequent itemsets, can association rules be derived by considering all permutations of items in a frequent itemset. Rules that satisfies both a minimum support threshold and a minimum confidence threshold are called *strong* [37].

The ideas of A-priori have been well proven in practical applications, and extended upon for various purposes. As mentioned by several authors (e.g. [34] and [23]), two main problems exist for association rule mining: 1) Low algorithm performance. 2) Results include a vast number of non-interesting rules. First, the theoretical number of rules grow exponentially with $|\mathcal{I}|$. Second, the number of possible rules generated from the final set of frequent items can be quite large, and even if it is not, finding truly interesting information is a non-trivial task. This problem introduces the problem of deriving robust and usable interestingness measures for association rules, in order to identify the truly interesting rules in practical applications.

Another measure commonly used together with association rules is *lift* (first introduced as *interest* by Brin, et al. [9]) which is defined as

$$lift(X \rightarrow Y) = \frac{conf(X \rightarrow Y)}{supp(Y)} \tag{2.3}$$

A lift value greater than 1 indicates an increase in probability of the consequent given the antecedent (itemsets containing X tend to contain Y more often than itemsets that do not contain X), while a lift value smaller than 1 indicates a negative correlation between the items. If the lift value is (near) 1 then the items appear together as often as expected due to random chance. Lift is further discussed in Chapter 5.

2.2 Information theory

Information theory is generally considered to have been founded in 1948 by Shannon in his ground breaking work, "A Mathematical Theory of Communication" [40]. Information theory was intentionally

[1]The term *large itemset* was originally used by [1]

developed for communication systems, but has become an important element in a variety of theoretical fields. Information theory is a branch of the mathematical theory of probability and statistics. As such, its abstract formulations are applicable to any probabilistic or statistical system of observations [26].

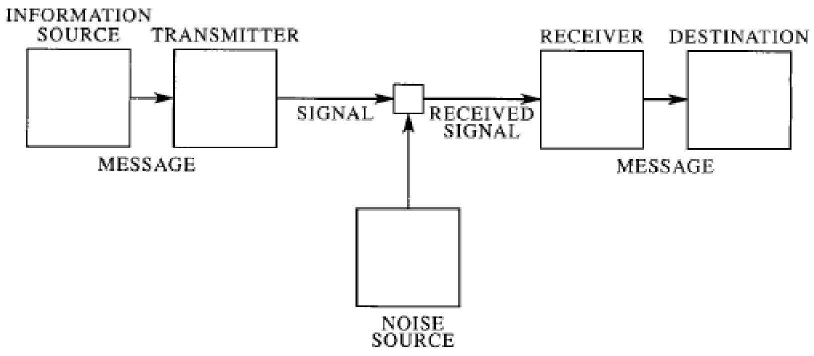


Figure 2.1: Schematic diagram of a general communication system

Information theory concerns the transmission of information on a discrete channel, see Figure 2.1. Generally, a discrete channel will mean a system whereby a sequence of choices from a finite set of elementary symbols S_1,...,S_n can be transmitted from one point to another. The source is generating the message, symbol by symbol. It will choose successive symbols according to certain probabilities, depending, in general, on preceding choices as well as the particular symbols in question. [40] This process can be defined as a Markoff process. Suppose we have a set of possible events,whose probabilities of occurrence are p_1,p_2,...,p_n. These probabilities are known, but that is all we know concerning which event will occur. The basic measure in information theory is a measure of how much "choice" which is involved in the selection of the event, or how uncertain we are of the outcome. This measure is called *entropy* and is defined as [40]:

$$H = K \sum_{i=1}^{n} p_i \log p_i \tag{2.4}$$

where K is a positive constant and p_i is the probability of the outcome i of possible outcomes generated by the source. The entropy in the case of two possibilities (binary) with probabilities p and $q = 1 - p$ is plotted in Figure 2.2.

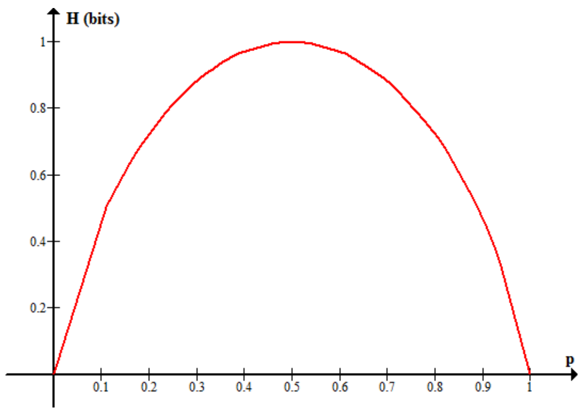


Figure 2.2: Entropy in the case of two outcomes with probabilities p and $(1 - p)$

The quantity H has a number of interesting properties which further substantiate it as a reasonable measure of choice for information [40]:

1. $H = 0$ if and only if all the p_i but one are zero, this one having the value unity. Thus only when we are certain of the outcome does H vanish. Otherwise H is positive.
2. For a given n, H is maximum, and equal to log$n$ when all the p_i are equal (i.e., $\frac{1}{n}$). This is also intuitively the most uncertain situation.

Entropy measures the number of bits needed to encode and transmit the outcome of random variables. The more certain an outcome of a random variable is, the less number of bits is needed to encode and transmit the information. If the outcome of a random variable is certain, there is no need to transmit any information, because the receiver already knows the answer, hence 0 bits is required to transmit the message. Consider the example of transmitting the message ABBO, where the literals A,B and O have the probabilities $\frac{1}{4}$,$\frac{1}{10}$,$\frac{1}{6}$ of occurring in the language. The minimum number of bits needed to encode and transmit the message then becomes:

$H(\frac{1}{4}, \frac{1}{10}, \frac{1}{10}, \frac{1}{6}) = -(\frac{1}{4}log(\frac{1}{4}) + \frac{1}{10}log(\frac{1}{10}) + \frac{1}{10}log(\frac{1}{10}) + \frac{1}{6}log(\frac{1}{6})) = 11,23$ bits

From the entropy definition, the *joint entropy* and *conditional entropy* can be derived. Suppose there are two events, x and y, in question with m possibilities for the first and n for the second. Let $p(i, j)$ be the probability of the joint occurrence of i for the first and j for the second. The entropy of the joint event is [40]:

$$H(x, y) = -\sum_{i,j} p(i, j) \log p(i, j) \tag{2.5}$$

Conditional entropy of y given x, $H(y|x)$, is defined as the average of the entropy of y, for each value of x, weighted according to the probability of getting that particular x [40]. That is:

$$H(y|x) = -\sum_{i,j} p(i, j) \log p(j|i) \tag{2.6}$$

This quantity measures how uncertain we are of y on the average, when we know x. Conditional entropy is particular useful for measuring the information x provides about y, and can be used as a foundation for assessing the interestingness of association rules. Interestingness measures for association rules based on information theory is the topic of Chapter 5.1.3.

Chapter 3

Rule syntax and semantics

Standard association rule mining has a well proven and clearly defined rule syntax and rule semantics. As described in Chapter 2.1, these rules are suited for market basket analysis where the data is transaction based. This thesis considers mining association rules from temporally ordered event sequences, with an additional requirement that they should be suited for prediction of failures in oil production. This specific application introduces some new requirements, and a new set of possibilities. One important aspect is the formalism needed to handle sequences of events instead of transactions, which is a well know problem discussed by Manilla et al. [30, 29, 31]. In their work Manilla et al. introduce the novel concept of episode rules as a framework for mining association rules from event sequences. A second important aspect is that the form of rules suited for prediction can be explicitly targeted by using a restricted rule syntax and semantics. This second aspect is highly domain specific, and sparsely covered by the literature in the field. Hence, some choices and assumptions must be taken while characterizing specific properties of rules suited for prediction.

This chapter will first introduce the details of association rules and episode rules, before the desired rule syntax and rule semantics are defined. At last, some implementation concerns with respect to the chosen type of rules in the practical setting of the ConocoPhillips project will be discussed.

3.1 Existing rule types

In the literature there exist two dominant rule types; association rules and episode rules. Association rules are the standard rule syntax for market basket data, while episode rules can be seen as an extension of association rules suited for event sequences. Both of them are important in this work as a reference, while defining rules suited for prediction of time based events.

3.1.1 Association rules

Association rules, as described in Chapter 2.1, provides a clear and intuitive rule syntax and rule semantics for market basket data. An association rule is on the form $X \rightarrow Y$ where X and Y are itemsets. The left-hand side (X) of the rule is called the $antecedent$, while the right-hand side (Y) is called the $consequent$. An association rule is interpreted as an association between the elements on the left hand side of the rule and the elements on the right hand side, stating that they frequently occur together in the same transaction. If the elements on the left hand side appear in a transaction the elements on the right hand side are likely to appear within the same transaction with a given probability (confidence).

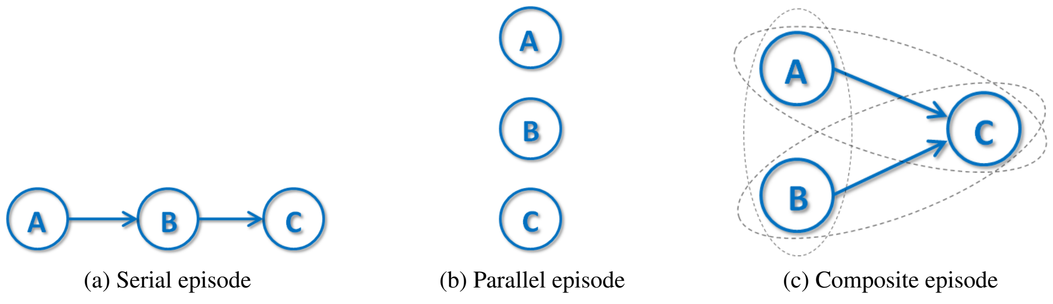
This rule syntax and rule semantics is the basis for this thesis. However, regular association rules is restricted to market basket data which consists of transactions. In the domain at hand the data is on the form of an event sequence, and there exists no notion of transactions. Hence, association rules must be extended to handle event sequences and a notion of time.

3.1.2 Episode rules

Specialized procedures have been developed for event sequences, providing a broader framework and expression power of the resulting rules. One particular popular framework is episode rules, which is an extension of association rules, introduced by Manilla et al. [30, 31]. Their approach incorporates the mechanisms necessary to extend transactional association rule mining to time series through the concept of episodes.

Mannila et al. [30, 31] consider the input as a sequence of events, where each event has an associated time of occurrence. Formally, the data is a set R of *events*, where every *event* is a pair (A, t) where $A \in R$ is the *event type* and t is an integer, the *time of occurrence* of the event. An *event sequence* **s** on R is a triple (s, T_s, T_e) where T_s is the starting time and T_e is the ending time of the sequence, $T_s \leq T_e$ are integers, $s = \langle (A_1, t_1), (A_2, t_2), ..., (A_n, t_n) \rangle$ and $A_i \in R$ and $T_s \leq t_i \leq T_e$ for all $i = 1, ..., n$.

One basic problem in analysing such sequences is to find frequent episodes. An episode is a collection of events occurring frequently close to each other. Episodes, in general, are partially ordered sets of events. An episode can be either serial or parallel, as illustrated in Figure 3.1. Formally, an episode is a pair $(V, \leq)$ where V is a collection of event types and $\leq$ is a partial order on V. Given a sequence S of alarms, an episode $a = (V, \leq)$ occurs within S if there is a way of satisfying the event types in V using the events of S so that the partial order $\leq$ is respected. This definition is recursively decomposable, and allows for composite episodes consisting of e.g. two episodes. The episode is parallel if the partial order relation $\leq$ is trivial. The episode is serial if the partial order relation $\leq$ is a total order.



(a) Serial episode (b) Parallel episode (c) Composite episode

Figure 3.1: Example of serial, parallel and composite episode types

Episode rules give the required framework needed to handle rule mining in event sequences. Episode rules also introduce a notion of time in the rules. This is done in two different ways based on how the episode rules are found. The two approaches are the WINEPI rule syntax and the MINEPI rule syntax. The WINEPI algorithm [30] defines a rule syntax that gives information about *total rule time*, which is the time all events included in the rule must occur within. The MINEPI rule syntax [29] gives information about both *antecedent time* and *total rule time*. Where *antecedent time* is the time the events included in the antecedent (left-hand side) of the rule must occur within.

WINEPI episode rules are interpreted as regular association rules, but with an additional time aspect, and the notion of episodes. If events satisfying the rule antecedent (left-hand side) occur in the right order within w time units, then also the rule consequent (right-hand side) occurs in the location described by $\leq$, also within w time units.

For example, we can have a rule

IF	high temperature
	high pressure
THEN	production stop
WITH	[300] conf(0.82) freq(580/2154)

which tells us that in 82% of the cases, where high temperature and high pressure occurred within 300 seconds, production stop also occurred within the same 300 seconds.

The MINEPI rule syntax [29] gives the ability to find different rules based on the time between the occurrences in the antecedent. E.g. if the events A and B occur within 10 minutes this results in C, but if they occur within 45 minutes they will most likely result in Y. A MINEPI episode rule gives the observed conditional probability that a certain combination of events occurs within some time bound, given that another combination of events has occurred within a time bound. Besides this, are MINEPI rules in general interpreted in the same way as WINEPI rules.

For example, we can have a rule

IF	high temperature
	high pressure
THEN	production stop
WITH	[50] [300] conf(0.82) freq(580/2154)

which tells us that in 82% of cases, where high temperature and high pressure occurred within 50 seconds, production stop occurred within 300 seconds

Episode rules introduce an important framework, but still have shortcomings with respect to the oil production domain. The purpose of this thesis is to find rules which are suited for prediction of time based events. Episode rules are well suited for event sequences, giving information about associations occurring close in time, but do not emphasise that the mined rules should be suited for prediction.

3.2 Suggested rule syntax

Rules suited for prediction of time based events differ from regular association rules and episode rules. One important difference is the need for a more restricted form of rule syntax, ensuring that the form of rules that are unwanted for prediction are not allowed. Another important difference is the semantics for interpreting the rules, and time of occurrence of events, with respect to prediction. Issues regarding semantics are the topic of Chapter 3.3.

When combining rules in a reasoning structure, the rule syntax becomes important in order to build a sound reasoning structure. Using the standard rule syntax and semantics of association rules and episode rules as foundation this chapter will define a rule syntax suited for prediction of time based events. The rule syntax can be divided into three components: Rule type, information about statistical strength of a rule and information about time in a rule. Each of these areas will be outlined in the following sub-chapters.

3.2.1 Rule type

The type, or structure, of discovered rules becomes important when they are to be applied in a reasoning system because some rule structures make the construction of a reasoning system difficult. First, recall from Chapter 3.1.1 that an association rule, $A \rightarrow B$, consists of two parts: the antecedent (left side, A) and the consequent (right side, B). This gives four combinations of number (single or multiple) of events in the antecedent and the consequent, as described in Table 3.1.

Rule structure	**Example**
single → single	A → B
single → multiple	A → B,C
multiple → single	A,B → C
multiple → multiple	A,B → C,D

Table 3.1: Possible association rule structures

Whereas the standard association rule syntax and episode rule syntax allows for any number of elements in both the antecedent and the consequent, Helland and Christiansen states in [11] that the best suited rule type for integration in a following reasoning structure is rules on the form: $(multiple \rightarrow single)$. Rules with multiple events on the consequent side can be redundant, because the reasoning structure will end in a single target event, which is the event interesting to predict. The other option, rules with only one event on the consequent side makes it easy to build a reasoning chain that ends in one target event, without predicting other non-interesting events. Hence, a rule structure with a single event in the consequent is preferred.

Rules with a single event in the antecedent does not support combinations of events as a trigger, while rules that allows for multiple events in the antecedent makes it possible to find combination of events that together contribute to a failure. In the ConocoPhillips case, which is a process system consisting of a large number of parallel processes, events can occur in all sorts of combinations. This thesis will make the following hypothesis:

Hypothesis 1: *Combinations of events greatly affect how the system operates, and interesting rules for predicting failures are found by looking at combinations of low level events.*

Given the hypothesis it is necessary to be able to find rules that combine trigger events, and a multiple antecedent is required. Hence, the preferred choice of rule structure in this thesis is rules on the form: $(multiple \rightarrow single)$.

3.2.2 Information about strength

Standard association rule syntax include statistics about a rule in the form of support and confidence. As stated in [43] the complete contingency table for a rule can be derived from the three measures

$support$, $confidence$ and $lift$ [9]. One of the key issues of this thesis is to identify those rules that are interesting with respect to prediction. This is done by calculating interestingness measures (Chapter 5.1) which operates on some part of the 2 x 2 contingency table for a rule. Hence, in order to provide the information required to compute the various interestingness measures, $support$, $confidence$ and $lift$ will be included in the recommended rule syntax.

3.2.3 Information about time

The last aspect of the rule syntax is the amount of time information included. It can be argued that the rule syntax should include as much information about the time distribution of events in a rule as possible, or the contrary, that such information is unnecessary. Considering the WINEPI and MINEPI rule syntax for time, both provides a valid syntax, but their definition of time may not be correct with respect to prediction.

The WINEPI rule time [30] gives the maximum total rule time, including both the antecedent and the consequent, which gives no information about the time from the observation of the antecedent to the consequent. For an operator such information can be important, because information about the time span of a rule can influence the decision making. Only considering the rule syntax, the rule time can be defined and found in other ways. One possible approach is to evaluate the occurrences of all instances of a rule, and use the maximum time between the antecedent and consequent. Another approach is to give some statistics about each rule with respect to time, e.g. minimum value, maximum value, mean and standard deviation of the occurrences of the rule. Such information can be presented to an operator in an explanation sequence, providing an indication on the timespan of the warning. To make the "interval" defined by these values more compact outliers can be removed, and a time interval which e.g. describe 95% of the occurrences of the rule can be given.

The MINEPI rule syntax [29] gives the ability to find different rules based on the time between the occurrences in the antecedent. E.g. if the events A and B occur within 10 minutes this results in C, but if they occur within 45 minutes it will most likely result in Y. Such differences can prove to be important, but this thesis will make the following hypothesis:

Hypothesis 2: *The time between occurrences of events in the antecedent, inside a reasonable interval, will not influence the outcome (consequent) of a rule.*

Based on this hypothesis, and an understanding of the work done in [20], where MINEPI rules will make it difficult to handle occurrences of events, the MINEPI rule syntax is evaluated as not necessary for predicting failures in the oil production domain.

Whether information about time is included in the rule syntax should be carefully evaluated according to if the information gives additional value to the reasoning process or to the end user. For example: If the user is presented with a warning saying that a failure may happen with a given probability within 2,5 hours, this can be misinterpreted by the user. The additional information about time can give the user an illusion of "false safety" if it is interpreted as a guideline on when the fault may occur. Given the information available, it may be equally likely that the fault event can happen within the first 10 minutes. Hence it may be better to exclude the time syntax, or expanding it to include e.g. minimum time, expected time and maximum time, instead of only maximum time. This thesis takes the last standpoint, recommending a rule syntax including information about the minimum time, expected time and maximum time between the antecedent and the consequent.

3.3 Suggested rule semantics

As already introduced, the rule semantics is important for rules which will be used to predict time based events. The handling of time in a reasoning structure relies on a clear interpretation of the rules, and how they relate to the time of occurrences of events. This interpretation of rules is defined in the rule semantics, which is the topic of this sub-chapter. The semantics can be divided into two main areas; ordering among events and defining a notion of time. Both aspects will be discussed in the following sub-chapters.

3.3.1 Ordering among events

The ordering among events becomes important when handling time based events. A rule which consists of events can impose various temporal orderings on the events. The imposed temporal ordering needs to be defined, and evaluated with respect to the domain at hand. Given a rule structure on the form $(multiple \rightarrow single)$, the required temporal order of events imposed by a rule becomes important in two ways:

1. The order imposed between the antecedent and the consequent.
2. The order imposed among the events in the antecedent side of the rule.

This gives four possible combinations of ordering among events in the case of a single consequent, as seen in Table 3.2.

Order between antecedent and consequent	**Order in the antecedent**
parallel	parallel
parallel	serial
serial	parallel
serial	serial

Table 3.2: Possible ordering of events

There exist two options for the order imposed between the antecedent and the consequent:

1. **Parallel:** The rule imposes no order between the antecedent and the consequent.
2. **Serial:** The rule imposes a total order on the events of the antecedent and the consequent, requiring that events in the antecedent occur before, with respect to time, the consequent.

As an example, consider the rules $A \rightarrow B$ and $B \rightarrow A$. With a serial rule these two rules differ, in opposite to a parallel ordering where they constitute the same rule. Classical association rule mining algorithms, such as A-priori, impose no temporal order among events because their goal is to describe association within transactions, hence they will give parallel rules. E.g. that bread is often bought in combination with milk, where the temporal ordering does not matter.

Considering the case of predicting failures in oil production, which is an ongoing time-spanning process, it is only interesting to predict events forward in time. Hence, the second option of serial rules, imposing a total temporal order between the antecedent and the consequent is better suited for selecting the right rules for prediction. Introducing this requirement in practice has complications, because it requires custom implementation of the rule mining algorithm. In other words, it will not be possible to use A-priori, without customization, to mine rules of this form. However, there exist many implementations of sequence rule mining [3], which is the case of serial ordering between antecedent and consequent, and a serial antecedent. Different rule mining algorithms is the topic of Chapter 4.

The second important issue is the order imposed by a rule among the events in the antecedent. As seen in Table 3.2 rules can have a serial or parallel antecedent. As an example, consider the rules $A, B \rightarrow C$ and $B, A \rightarrow C$. With a serial antecedent these two rules differ, in opposition to if the antecedent is parallel. The consequence of using a parallel antecedent is that it is not possible to detect rules where the ordering of trigger events is important. In a process system it may be that only a specific ordering of events will cause a failure, while the same events in some other ordering will not. Hence, the choice of parallel versus serial antecedent becomes highly domain specific, and should be motivated by the following question: Is the specific ordering of events the important factor in triggering failures, or is the occurrences of some events inside a given time interval sufficient to produce a failure?

According to Maintenance Optimisation Engineer Fredrik Høymer Fossan at ConocoPhillips, the relations interesting to predict mainly corresponds to the occurrence of certain events within a specific time interval [15]. Based on this, the target for the rule mining process in ConocoPhillips is rules on the form $[serial, parallel]$, but it should be noted that important information about process development can be lost by only considering a parallel antecedent.

3.3.2 Notion of time

The information a rule can give about time can be divided into three main areas:

1. Maximum time of validity.
2. Minimum time of validity due to a minimum reaction time for operators.
3. The expected timespan of a rule.

Each area will be introduced in the following sub-sections, where possible solutions will be outlined and discussed.

Maximum time of validity

Prediction of time based events implies that events have an associated time of occurrence. The question of how to handle these occurrences in mined rules is essential, because a rule can only be valid during some time interval. In contrast to regular association rule mining, where a rule has a clearly defined area of validity inside a transaction, rules mined from event sequences do not have such a predefined area of validity. In general, may a rule mined from an event sequence be valid across the whole data set, yielding associations with a time span of the whole data set, perhaps months or years. Clearly, associations of such a time span will not be suited for prediction, and will be highly sensitive to noise. However, in a process system the rules can be constrained to an interval as large as the maximum likely time of influence between events, as defined by a domain expert. Rules mined using a sliding window approach (the topic of Chapter 4.1.1), will be defined as valid inside the interval defined by the length of the sliding window. Hence, the size of the sliding window should be equal to the maximum likely time of influence between events.

The interval of validity for a rule is also important with respect to the following reasoning structure. As discussed in [20], the reasoning structure needs a formalism for how to state "evidence", and how to handle this "evidence" as time elapses. If an event happens in real-time, this is stated as "evidence" for this event in the reasoning structure. The problem is how to handle this evidence after it has been stated. At some point this evidence must be removed when the validity of the evidence has elapsed. If all rules have a defined interval of maximum validity, [20] states that the evidence can safely be removed when this interval has elapsed, because the maximum time of influence from this event on other events has elapsed without anything happening. This thesis will assume that a maximum likely time of influence between events can be derived by a domain expert, and that this defines the maximum time of validity for a rule.

Minimum time of validity

Another important aspect of time validity is obtained by introducing a lower bound on rule validity, based on the reaction time of operators. In a practical setting, rules with an expected time of occurrence below some threshold, is of little use to an operator because he or she will not be able to utilize the knowledge before the consequent already has happened. Hence, rules with an expected time of occurrence below the reaction time of operators may be removed, in order to reduce the number of rules. If this should be done, a criterion for determining the expected time of a rule is needed. This problem corresponds to the next section, and will not be discussed here. This thesis chooses to use a minimum time of validity defined by experts as a threshold for all rules. If the expected time of occurrence of a rule is below the defined minimum reaction time for the operators, then the rule will be discarded because it will not help the operators to avoid a possible failure.

Expected timespan of a rule

Independent of which information to include in a rule, a definition of how the time between the antecedent and consequent is found is required. Some alternatives include using the time between the first element of the antecedent and the consequent, using the time between the last element of the antecedent and the consequent, or using the time between the midpoint ($\frac{first+last}{2}$) of the antecedent and the consequent. These alternatives are illustrated in Figure 3.2. Based on easy calculation of time values, and easy maintenance of occurrences in a reasoning structure, the preferred choice for this thesis is to use the time between the last element of the antecedent, and the consequent. This is also an intuitive interpretation for an user, because one would expect the time of a rule to be interpreted as the time from the antecedent events are observed to the effect of the rule occur.

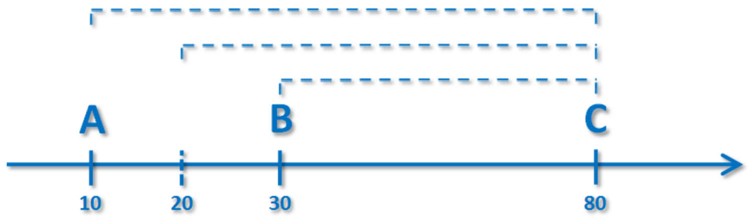


Figure 3.2: Three alternatives for definition of time between antecedent and consequent.

3.3.3 Summary of rule syntax and rule semantics

The recommended rule syntax and rule semantics, for rules suited for prediction of time based events, is given in Table 3.3. It is important to be aware that all elements are a result of choices made in this thesis, and will affect the result in the end. The consequences of these choices will be discussed in Chapter 6.1. The rule syntax and rule semantics defined here will be the underlying foundation for the theoretical rule mining framework presented in the following chapters.

Rule type	$multiple \rightarrow single$
Time information	$minimum$, $average$ and $maximum$ time between the last event in the antecedent and the consequent event.
Statistics	$support$, $confidence$ and $lift$.
Rule ordering	$parallel$ antecedent, $serial$ rule
Maximum time of validity	Maximum likely time of influence between events.
Minimum time of validity	Operators reaction time.
Example	$A, B \rightarrow C$ *(min: 10, avg: 30, max: 60)* *(sup: 7%, conf: 90%, lift: 7)*

Table 3.3: Rule syntax and rule semantics for prediction of time based events.

3.4 Implementation concerns

In the ConocoPhillips project, which this study is a part of, data mining software from SAS Institute is preferred. Hence it is relevant to explore the capabilities of the SAS Enterpriser Miner 5.2 software, with respect to the ability of handling the rule types discussed. SAS Enterprise Miner 5.2 provides a powerful and complete data mining solution with unparalleled model development and deployment alternatives and extensive integration opportunities [24]. For doing market basket analysis SAS Enterprise Miner 5.2 supports association and sequence discovery, and supports measures such as *support*, *confidence* and *lift* [24].

SAS Enterprise Miner 5.2 supports different structures of rules in the two different rule mining paradigms; association discovery and sequence discovery. In association discovery, all rules found are on the form $(multiple \rightarrow multiple)$. However, doing a second pass over the rule set removing the rules with more than one element in the consequent is easy. In sequence discovery the rules are given as sequences of events (e.g. $A \rightarrow B \rightarrow C$), but by combining all events excluding the last event into an antecedent, gives rules on the form $(multiple \rightarrow single)$ (e.g. $A, B \rightarrow C$). Hence, SAS Enterprise Miner 5.2 supports the rule structure chosen in this thesis, and allows for easy implementation.

Considering the four possible combinations of rules in Table 3.2, association discovery in SAS Enterprise Miner 5.2 results in rules of the form $[parallel, parallel]$, and sequence discovery results in rules of the form $[serial, serial]$. Hence, no standard procedure in SAS Enterprise Miner 5.2 gives the wanted ordering among events in the rule.

However, there exist two possible solutions to obtaining rules with the wanted ordering in Enterprise Miner. One possible solution is to mine $[parallel, parallel]$ rules using the association discovery procedure, and do another pass over the data to count occurrences satisfying the temporal order. Another solution is to use the support and confidence of each sequence rule ($[serial, serial]$), and combine sequence rules containing the same items in the antecedent into rules of the form $[serial, parallel]$. To see this, consider the following example:

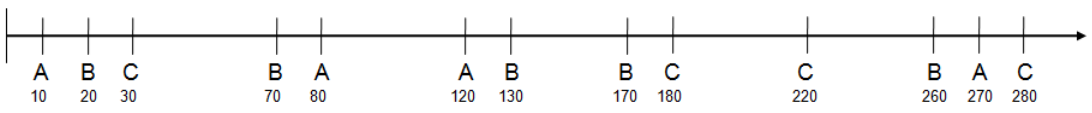


Figure 3.3: Event sequence example

Given the events A, B, C occurring along a time line, as described in Figure 3.3, and their respective time of occurrence given in minutes noted below each occurrence. Using a sliding window algorithm, as described in Chapter 4.1.1, with a window width of 30 minutes ($[0-30>$), and a step size of 10 minutes, gives 29 "transactions" which is the basis for both an association discovery and a sequence discovery. Performing these tasks in SAS Enterprise Miner 5.2 gives the results in Figure 3.4.

Considering the results from Figure 3.4, it is possible to derive rules of the form $[serial, parallel]$ from the sequence rules. The rules $A \rightarrow B \rightarrow C$ and $B \rightarrow A \rightarrow C$ can be written as $A, B \rightarrow C$ and $B, A \rightarrow C$ respectively. Both rules are of the form $[serial, serial]$. These two rules constitute the only combinations of orders of A and B in the antecedent, and by using the support and confidence of each rule we can derive the rule $A, B \rightarrow C$ on the form $[serial, parallel]$. This can be done in the following way:

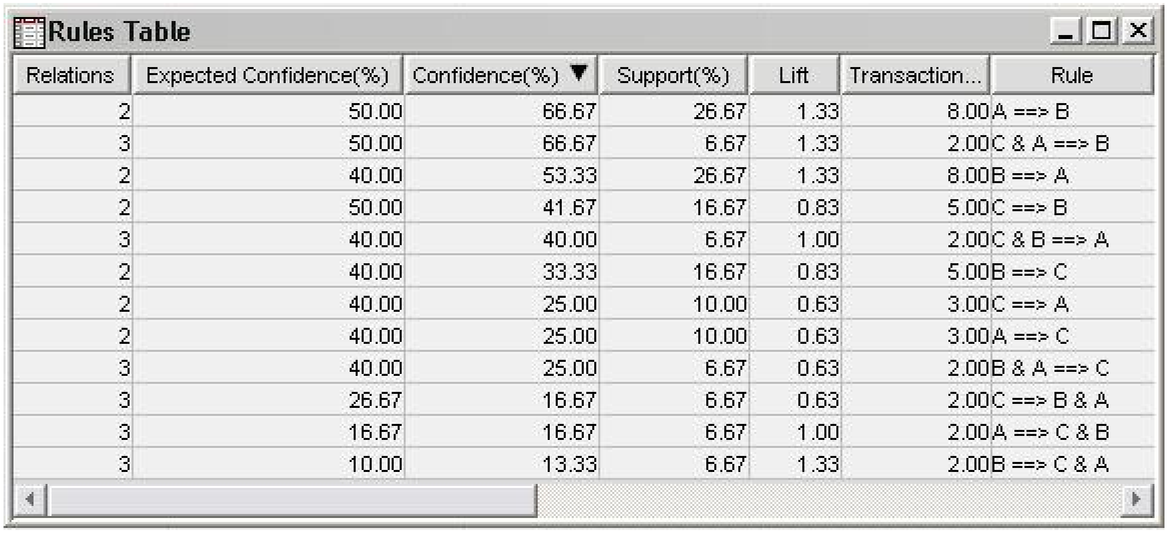
Recall the definition of confidence from Chapter 2.1. Support of the sequence $A \rightarrow B \rightarrow C$ is 1. Then the support of the sequence A, B is

$$support(A, B) = \frac{support(A \rightarrow B \rightarrow C)}{confidence(A \rightarrow B \rightarrow C)} = \frac{1}{0.25} = 4$$

Support of the sequence $B \rightarrow A \rightarrow C$ is 1. Then the support of the sequence B, A is

$$support(B, A) = \frac{support(B \rightarrow A \rightarrow C)}{confidence(B \rightarrow A \rightarrow C)} = \frac{1}{0.25} = 4$$

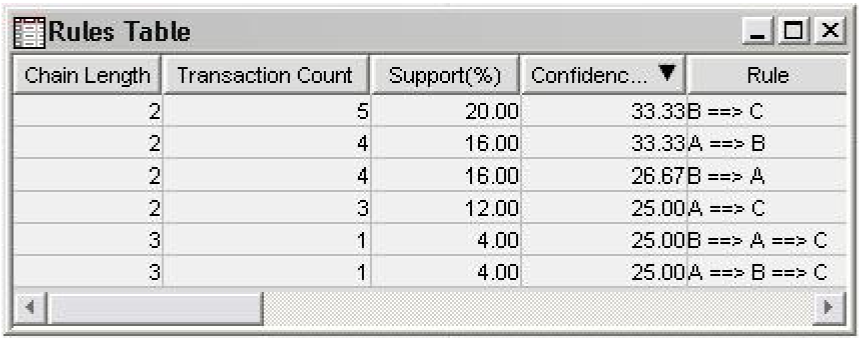
The support of the $[serial, parallel]$ rule $A, B \rightarrow C$ is found by adding the support of the possible combinations of orders in the antecedent, giving: $support(A, B, C) = 1 + 1 = 2$. Now, the confidence of the $[serial, parallel]$ rule $A, B \rightarrow C$ can be found by



Rules Table

Relations	Expected Confidence(%)	Confidence(%) ▼	Support(%)	Lift	Transaction...	Rule
2	50.00	66.67	26.67	1.33	8.00	A ==> B
3	50.00	66.67	6.67	1.33	2.00	C & A ==> B
2	40.00	53.33	26.67	1.33	8.00	B ==> A
2	50.00	41.67	16.67	0.83	5.00	C ==> B
3	40.00	40.00	6.67	1.00	2.00	C & B ==> A
2	40.00	33.33	16.67	0.83	5.00	B ==> C
2	40.00	25.00	10.00	0.63	3.00	C ==> A
2	40.00	25.00	10.00	0.63	3.00	A ==> C
3	40.00	25.00	6.67	0.63	2.00	B & A ==> C
3	26.67	16.67	6.67	0.63	2.00	C ==> B & A
3	16.67	16.67	6.67	1.00	2.00	A ==> C & B
3	10.00	13.33	6.67	1.33	2.00	B ==> C & A

(a) Association discovery



Rules Table

Chain Length	Transaction Count	Support(%)	Confidenc... ▼	Rule
2	5	20.00	33.33	B ==> C
2	4	16.00	33.33	A ==> B
2	4	16.00	26.67	B ==> A
2	3	12.00	25.00	A ==> C
3	1	4.00	25.00	B ==> A ==> C
3	1	4.00	25.00	A ==> B ==> C

(b) Sequence discovery

Figure 3.4: Screenshots of results from SAS Enterprise Miner 5.2

$$confidence(A, B \rightarrow C) = \frac{support(A, B, C)}{support(A, B)} = \frac{1 + 1}{4 + 4} = 0.25$$

The two combinations of $[serial, serial]$ rules have been transformed into one $[serial, parallel]$ rule with $support = 2$ and $confidence = 0.25$, solely based on the information found in each rule, which is correct with respect to the 29 transactions derived by the sliding window algorithm. These two solutions show that it is possible to derive the wanted form of rules in SAS Enterprise Miner 5.2.

When it comes to time, SAS Enterprise Miner 5.2 has no support for time information in rules. However, it is possible to use the mined rules and do a second pass over the data to generate the wanted time information. As already introduced, the maximum time validity of a rule will be defined by the size of the sliding window, which should be defined as the maximum likely time of influence between events. The minimum time validity of rules can be enforced in SAS Enterprise Miner 5.2 by using the time information gathered in the second pass over the data, and discard all rules with a expected time of occurrence below the reaction time of operators.

The conclusion is that the defined rule syntax can be enforced in SAS Enterprise Miner 5.2 during a post-processing phase, but requires additional passes over the data to obtain the wanted time information and ordering. This additional post-processing phase is from a theoretical point of view not necessary, because the rule syntax suggested can easily be enforced during the rule mining phase. This motivates for specialized algorithms that respect the nature of event sequences and the syntax of rules suited for prediction.

Chapter 4

Rule mining

Association rule mining, as introduced in Chapter 2.1, provides a formalism for discovering information about association patterns in transaction databases. Use of the association rule mining paradigm for prediction of production failures in the oil production domain is motivated by the extended discussion in the recent project thesis by Christiansen & Helland [11]. Here two main conclusions are important: (1) Using an event based approach based on association rules for modelling irregular behaviour will probably perform better in predicting production failures than other methods suggested in the literature. (2) Mining rules based exclusively on support and confidence will result in a vast number of rules, and may fail to identify the truly interesting rules in the oil production domain, hence some other strategy for obtaining a more precise set of rules is needed.

The first conclusion implies that a robust way of finding association rules in event sequences is needed. The rule syntax and rule semantics defined in Chapter 3 provides a foundation for rules which can handle event sequences, and which are suited for prediction. The problem then becomes how to find such rules from a large sequence of time based events. The second conclusion implies two things: First, the resulting rule set from standard association rule mining algorithms is far too numerous to handle, indicating a underlying problem of the method. Second, a new problem of identifying interesting rules from a possibly large set of rules is introduced.

A theoretical framework for handling these problems is suggested. It can be considered as a two step process, consisting of rule mining and rule selection. Rule mining is the topic of this chapter, while rule selection is the topic of Chapter 5. With the defined rule syntax and rule semantics in mind two possibilities exists. They can be enforced as an integrated part of the rule mining phase, or during post-processing in the rule selection phase. This implies that the distribution of tasks into each phase of the framework is not fixed, as seen in this chapter.

This chapter will first introduce how rules can be mined from an event sequence. Next, the problem of how the discovery of a vast number of rules can be restricted by specialised algorithms in such a way that a smaller set of relevant rules is found, and performance and resource usage comes within feasible limits, is introduced. To the authors knowledge there exists no solution which focus on obtaining rules suited for prediction, hence a suggested approach for rule mining specific to the domain at hand will be derived. Finally, some concerns with respect to implementation of the suggested solution in ConocoPhillips will be discussed.

4.1 Mining rules from event sequences

Classical association rule mining concerns mining rules from transactional data, not time series of events, hence they are not directly applicable to the domain at hand. Several different approaches for mining association rules from event sequences have been investigated [30, 29, 13]. Essentially, these transform the sequential data into transactions by using a sliding window technique, or employ a specialized algorithm for event sequences. All algorithms introduced in this chapter use A-priori as the basis, and can be considered as extensions of the A-priori algorithm to event sequences.

Other algorithms, such as [3, 18, 46], exists in the literature for mining the restricted form of sequence rules. Sequence rule mining is more complex and computational demanding than association rule mining, and because the restricted form of sequence rules is not needed in the ConocoPhillips domain (see Chapter 3), these are omitted from this study.

4.1.1 Sliding window techniques

A common way to transform a continuous sequence of events into a transaction database is to divide the event sequence into smaller sub-sequences which become analogue to transactions, where e.g. standard A-priori based algorithms can be applied. The most common approach is to use a sliding window technique, where a fixed time window (time interval) is slided along the sequence. Events covered by the time interval constitute a transaction. The fixed size window (interval) is then moved one step further, and those events then covered constitute the next transaction. Transactions can be partially overlapping. See Figure 4.1 for an illustrating example. A sliding window approach requires the user to a-priori fix two numbers, the size of the sliding window and the step size. Both numbers influence the resulting number of transactions and the distribution of events to transactions, hence affecting which association rules that eventually are produced.

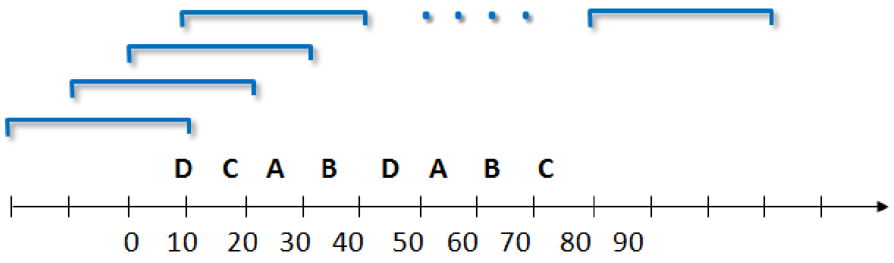


Figure 4.1: Sliding window extraction with width W=30 and step size S=10

The maximum timespan of a rule is equal to the size of the sliding window. Using standard association rule mining algorithms, the timespan of all rules will be equal or less than the sliding window size. The step size will directly influence the number of transactions created. A smaller step size will result in transactions with greater overlap, and hence produce more transactions. On the other side, will the step size determine the fairness of the transaction extraction procedure. A large step size will create possibly unfair limits between transactions, and perhaps divide important sequence combinations into two different transactions, which may result in inability to detect these associations. Hence the larger the step size, the more the result is influenced by chance, and vice versa. One important consideration is that a small step size for a large event sequence can produce a dataset larger than what is possible to handle in reasonable time with association rule mining algorithms, hence the chosen step size needs to be evaluated with respect to the computational resources available, and the required degree of fairness in the extracted transactions.

It is possible to use sliding window techniques in two ways; as an integrated part of the rule mining algorithm (such as in WINEPI), or as a standalone technique for transforming an event sequence to a transaction database. Without introducing the specifics for handling event sequences, and the rule syntax outlined in Chapter 3, A-priori can be used to mine standard (parallel, parallel) association rules. The problem of using this approach is that the rule syntax is not naturally enforced, and the rule mining algorithms produce a vast number of unwanted rules which must be pre-processed to produce the wanted rule syntax. Combined with the large amount of transactions produced by the sliding window technique, this results in an artificially large use of computing resources and running times on standard computers. The other way to approach the problem is to use specialised algorithms for mining rules from event sequences, which is the topic of the reminder of this chapter.

4.1.2 WINEPI

As already introduced, WINEPI [30, 31] is a specialised algorithm for mining episode rules, based on a sliding window approach. WINEPI utilizes the concept of episodes described in Chapter 3.1.2, which may be parallel, serial or composite. The idea is that to be considered interesting the events of an episode must occur close in time. The user defines how closely events must occur by specifying a *time window* and a *step size* defining the overlap of windows. The goal is the same as in regular association rule mining, to find all frequent episodes. This is done by counting in how many windows an episode occurs. If an episode a occurs in **S**, we write $\mathbf{S} \models a$. The frequency of a in the set $aw(\mathbf{S}, w)$ of all windows on **S** of size w is

$$fr(a, \mathbf{S}, w) = \frac{|\{W \in aw(\mathbf{S}, w) | W \models a\}|}{|aw(\mathbf{S}, w)|}.$$

Given a threshold min_fr for the frequency, a given episode a is frequent if $fr(a, \mathbf{S}, w) \geq min\_fr$. If an episode a is frequent in a sequence **S**, then all sub-episodes $\beta \prec a$ are frequent, as given by the A-priori property. The algorithm follows the same procedure as the A-priori algorithm described in Chapter 2.1, iteratively alternating between building and recognition phases. First, in the building phase of an iteration i, a collection C_i of new candidate episodes of i elementary events is built, by joining the frequent episodes that share the $i-1$ events. This can be done efficiently by representing an episode as a lexicographically sorted array of event types. Collections of episodes are also represented as lexicographically sorted arrays. Since the episodes and episode collections are sorted, all episodes that share the same first event types are consecutive in the episode collection. By storing the start point of blocks that share the $i-1$ events the candidate generation can be done efficiently.

Then, these candidate episodes are recognized in the event sequence, and their frequencies are computed. When episodes are recognized in sequences it is done in an incremental fashion, and the sliding window is applied on the event sequence. The episodes that are totally covered by the window are counted for each step of the window. Two adjacent windows are typically very similar to each other. This is used as an advantage by after recognizing episodes in W_i, incremental updates are made in the data structures to achieve the shift of the window to obtain W_{i+1}. Parallel episodes are recognized by maintaining a counter and a window list for each candidate episode, and serial episodes are recognized by using a state automaton. In addition it is possible to recognise all forms of composite episodes when counting, not only serial or parallel episodes. This can be done by handling the candidate episodes as regular parallel episodes until all events of the candidate episode is inside the window, and then check for the correct partial ordering. This gives the ability to only consider rules on the form $[parallel, serial]$, hence enforcing the rule syntax at runtime.

If $F(S, w, min\_fr)$ is the collection of frequent episodes in S with respect to w and min_fr, it is possible to derive all frequent episode rules from this collection. Formally, a WINEPI episode rule is an expression $\beta \Rightarrow \gamma$, where β and γ are episodes such that β is a sub-episode of γ and both are frequent. This will give rules with multiple events on the antecedent and consequent side, but it is easy to restrict the rule generation to only consider rules where the last element of the episode, which should be the last occurring event time wise, is the consequent. In addition to enforcing the wanted rule syntax this greatly reduce the running time of the algorithm.

The fraction $\frac{fr(\gamma,\mathbf{S},w)}{fr(\beta,\mathbf{S},w)}$ is the *confidence* of the episode rule. The confidence can be interpreted as the conditional probability of the whole of γ occurring in a window, given that β occurs in it. As already noted WINEPI can produce the wanted rule structure and ordering defined in Chapter 3. The semantics with respect to maximum time is also enforced naturally in the algorithm. However, WINEPI do not keep information about the occurrences of events, which makes it difficult to obtain the wanted time information in the rule syntax, and enforce the minimum time constraint.

4.1.3 MINEPI

MINEPI as described by Manilla et al. [29], [31] is another more recent algorithm for mining episode rules from sequential data. MINEPI differs from WINEPI at two important points; it uses the concept of minimal occurrence instead of a sliding window approach, and it gives the possibility of different time bounds for the antecedent part and the consequent part of the rule. MINEPI also incorporates a more generalized framework for episode discovery. The generalized framework allows one to express arbitrary unary conditions on the individual events, and to pose binary conditions on the pairs of events, giving the possibility to exactly target the types of event combinations that are interesting in the application [29]. An advantage of using minimal occurrences of episodes, besides being simple and efficient, is that confidence and frequencies of rules with different time bounds can be obtained quickly without the need to rerun the analysis for each time bound. More interesting for the purpose of prediction is that all occurrences of events and episodes are stored in the algorithm, which makes it easy to naturally enforce the rule syntax and semantics, and easily obtain the wanted time information for rules.

For simplicity the formalism complying with the generalized framework is omitted, and the procedure is described in the same terms as WINEPI. Doing this implies that we loose some generality by restricting the episodes to be parallel or serial, which is sufficient at this point. The complete specification of the MINEPI framework can be found in [29] and [31]. An episode $P(x_1, ..., x_k)$, where x are events, occurs in a sequence of events $\mathcal{S}= (e_1, ..., e_n)$, at interval $[t, t']$, if there are events $e_{j_1}, ..., e_{j_k}$ such that $P(e_{j_1}, ..., e_{j_k})$ is satisfied, and $t \leq min_i\{e_{j_i}\}$ and $t' \geq max_i\{e_{j_i}\}$. An occurrence of an episode P at $[t, t']$ is *minimal* if P does not occur at any proper subinterval $[u, u'] \subset [t, t']$. The set of (intervals of) *minimal occurrences* of an episode P is denoted by $mo(P)$ [29]:

$$mo(P) = \{[t, t'] | [t, t'] \textit{ is a minimal occurrence of P}\}.$$

The frequency $freq(P)$ of an episode P in a given event sequence $\mathcal{S}$ is defined as the number of minimal occurrences of P in the sequence $\mathcal{S}$, $freq(P) = |mo(P)|$. Given a frequency threshold min_fr, an episode P is frequent if $freq(P) \geq min\_fr$. The concept is the same as earlier, finding all frequent episode rules. The procedure follows the same steps as A-priori (and WINEPI), and is outlined in Algorithm 1. Having found the set L_k of frequent simple episodes of size k, the set C_{k+1} of candidate episodes of size $k+1$ is formed, by joining the frequent episodes that share the k first events. The set of candidate episodes is then evaluated by forming the set $mo(Q)$, where $Q \in C_{k+1}$, and checking if it really is frequent.

Algorithm 1: Discovery of frequent simple episodes [31]

```
Input: An event sequence $\mathcal{S}$, and a frequency threshold $min\_fr$
Output: All frequent simple episodes from $\mathcal{E}_{\mathcal{S}}$ (and their minimal occurrences)
1.  $C_1 :=$ the set of episodes of size 1 in $\mathcal{E}_{\mathcal{S}}$;
2.  forall $P \in C_1$ do compute $mo(P)$;
3.  $L_1 := \{P \in C_1 | P$ is frequent $\}$ ;
4.  $i := 1$;
5.  while $L_i \neq \emptyset$ do
6.      $C_{i+1} := \{P | P \in \mathcal{E}_{\mathcal{S}}$, all sub-episodes of $P$ are in $L_i\}$ ;
7.      $i := i + 1$;
8.      forall $P \in C_i$ do
9.          select sub-episodes $P_1$ and $P_2$;
10.         compute $mo(P)$ from $mo(P_1)$ and $mo(P_2)$;
11.     end
12.     $L_i := \{P \in C_i | P$ is frequent $\}$;
13. end
14. forall $i$ and all $P \in L_i$ do output $P$ and $mo(P)$ ;
```

For serial episodes the two sub-episodes are selected at line 9 so that P_1 contains all events except the last one and P_2 in turn contains all except the first one. The minimal occurrences of P are then computed on line 10 as: $mo(P) = \{[t, u'] \mid$ there are $[t, t'] \in mo(P_1)$ and $[u, u'] \in mo(P_2)$ such that $t < u$, $t' < u'$, and $[t, u']$ is minimal$\}$.

For parallel episodes, the sub-episodes are selected at line 9 so that P_1 and P_2 contain all events except one; the omitted events must be different. The minimal occurrences of P are then computed on line 10 as: $mo(P) = \{[v, v'] \mid [t, t'] \in mo(P_1)$ and $[u, u'] \in mo(P_2)$, $v = min\{t, u\}$ and $v' = max\{t', u'\}$, and $[v, v']$ is minimal$\}$.

As shown, minimal occurrences for both serial and parallel episodes of a candidate episode P can be found in a linear pass over the minimal occurrences of the selected sub-episodes P_1 and P_2. This makes the algorithm efficient, but the size of the data structures for holding the minimal occurrences may be even larger than the original database, especially in the first couple of iterations, making it hard to hold the structures in main memory for large databases.

From the frequent episodes, MINEPI episode rules can be derived. A MINEPI rule is an expression $\beta[win_1] \Rightarrow \alpha[win_2]$, where β and α are episodes such that β is a sub-episode of α, and win$_1$ and win$_2$ are integers. The interpretation of the rule is that if episode β has a minimal occurrence at $[t_s, t_e]$ with $t_e - t_s \leq win_1$, then the whole super-episode α occurs at interval $[t_s, t'_e]$ for some t'_e such that $t'_e - t_s \leq win_2$.

The *confidence* of the rule $\beta[win_1] \Rightarrow \alpha[win_2]$ is the conditional probability that α occurs, given that β

occurs, under the time constraints specified by the rule:

$$confidence(\beta \Rightarrow \alpha) = |mo(\alpha)|/|mo(\beta)|$$

where $|mo(\beta)|$ is the number of minimal occurrences $[t_s, t_e]$ of β such that $t_e - t_s \leq win_1$, and $|mo(\alpha)|$ is the number of such occurrences where there is also an occurrence of α within the interval $[t_s, t_s + win_2]$.

As already noted, MINEPI provides a simple and efficient framework for discovering episode rules. In addition it is easy to tailor the algorithm to naturally enforce the rule syntax and the rule semantics of Chapter 3. The concept of minimal occurrences also makes it easy to derive time information for rules. However, MINEPI provides no information about probability of events or episodes with respect to the whole database. Such probabilities are often needed, e.g. to compute $lift$ values for rules. In WINEPI the probability for an event occurring can easily be computed from the number of occurrences of the event, and the total number of possible sliding windows. In MINEPI there exists no notion of total possible windows or occurrences, hence the probability of an event or an episode is unknown. The way to solve this problem is to, as noted in [31], use MINEPI to solve the task of WINEPI in an additional computation. A window contains an occurrence of an episode exactly when it contains a minimal occurrence. The frequency of an episode α can thus be computed from $mo(\alpha)$ and the total number of windows can be computed, by setting the window size equal to the maximum time of MINEPI, and setting a window step size.

4.2 Rule mining in dense databases

Algorithms such as A-priori, WINEPI and MINEPI tend to produce a large number of rules, which often makes the problem of analysing and identifying interesting rules difficult. As explained in [39] the A-priori algorithm is developed for market basket data, which typically is a sparse database. Though the dimensionality is high in the data, the number of items in a record is tiny in comparison. This sparsity is exploited by algorithms based on A-priori. Datasets from other domains such as sensor data, and other classification and prediction tasks, in general tends to be dense in that they have the following properties [39]:

- many frequently occurring items
- strong correlations between several items
- many items in each record

Because of the dense data-set, resource consumption is exponentially boosted. In addition, is the number of rules mined large because the algorithms mine all rules satisfying a given minimum support threshold, some which are non-descriptive with regard to the prediction task. Recall from MINEPI, that rules are constructed by considering all frequent episodes α as the right-hand side, and all sub-episodes $\beta \prec \alpha$ as the left-hand side of the rule. With a frequency threshold which produce a large set of frequent episodes the number of rules quickly becomes vast. Tests done on the data from the ConocoPhillips case described in Chapter 1, results in approximately 443 000 rules with varying confidences using MINEPI. See Appendix A for details. To cope with the problem of a large number of non-descriptive rules, two techniques are introduced in the literature: 1) Enforcing other rule constraints during a post-processing phase. 2) Incorporate additional rule constraints in the rule mining algorithm, further enhancing the pruning of candidate itemsets and thereby the efficiency of the algorithm. Post-processing of rules is the topic of Chapter 5 of this report.

Two dominant techniques for incorporating rule constraints in rule mining algorithms exists in the literature: 1) Using constraints on the items allowed in a rule. 2) Incorporating other measures than the minimum support constraint in the rule mining algorithm to confine the search space. Both techniques will be outlined in the following sub-chapters.

4.2.1 Mining rules using item constraints

Srikant et al. [42] introduce a framework for enforcing constraints on the mined frequent itemsets. The method uses the candidate generation phase of A-priori as a basis, and is applicable to most A-priori based algorithms. The motivation is that users often are interested in only a subset of all associations, for instance those containing some items from a user-defined subset of items. While the output of standard A-priori algorithms can be filtered out in a post-processing step, it is much more efficient to incorporate such constraints into the rule mining algorithm. In the case of predicting a target event, this framework makes it possible to greatly focus the mining effort, and thereby increasing the efficiency of the algorithm.

In brief, the method lets the user specify that only rules containing some specific items are of interest, or contrary, rules with the absence of specific items. This constraint is enforced during the candidate generation phase of A-priori, where only candidates of the join operation that includes the specified items will be accepted. This will greatly reduce the number of candidate episodes. However, it is important to understand that enforcing such a constraint in a naive way breaks the completeness of the candidate generation process. Some candidates that are frequent will not be generated, hence not all frequent itemsets that adhere to the constraint will be generated.

For example, let the item constraint be that we want rules that contain the item 2, and let L_2 = { {1 2}, {2 3} }. For the A-priori join step to generate {1 2 3} as a candidate, both {1 2} and {1 3} must be present - but {1 3} does not contain 2 and will not be counted in the second pass. Even if all subsets ({1 2},{1,3},{2 3}) of {1 2 3} are frequent, and {1 2 3} contains 2, it will not be generated as a candidate because {1 3} was not generated as a candidate in the subsuming pass. The contribution of Srikant et al. is a solution to the problem of completeness, introducing various efficient ways to ensure that all frequent episodes that adhere to the item constraints are generated. The complete method will not be described in detail, but can be found in [42].

The method of Srikant et al. makes it possible to only target rules with a specific consequent. However, in the case of combining rules in a reasoning structure for predicting a target event, not only rules with the target as consequent are interesting. Clearly, rules with the target event in the consequent is needed for predicting the target event, but one also wants to predict the events which can lead to the target event. One possible approach is to develop a focused mining process which is run iteratively. First, all rules with the target as a consequent is mined using the item constraint. Next, the same procedure is repeated, with all events that appeared in the antecedent in the first step as the new item constraints. This process is repeated until a desired level of rules is reached. Such a process ensures that only rules which is directly applicable for prediction is found, but the performance benefits will only appear if the number of events in the item constraint, in the last level, is strictly lower than the total number of available events.

Preliminary tests on the ConocoPhillips data set show that, if the maximum number of iterations is 4 then 60 of 99 events will be included in the item constraints. At 5 iterations all events are included.

This indicates that a iterative focused mining process will receive little or no performance gain over standard association rule mining. The number of mined rules will also be in the same magnitude as in standard association rule mining. However, if the target event has a low support (below the lowest support threshold feasible with respect to running times of the algorithm), this method can be applied in order to make the discovery task of rules which include the target event feasible.

4.2.2 Mining rules using confidence and minimum improvement constraints

Bayardo et al. [39] outlines a constraint based rule mining algorithm for dense databases. This algorithm greatly improves running times for A-priori in dense databases, and just as importantly, it significantly reduces the number of non-descriptive rules mined. Their algorithm incorporates pruning of the search space (candidate itemsets) by two additional measures; a bound on minimum confidence and a bound on minimum improvement.

The minimum improvement constraint is based on the idea of mining only those rules whose confidence is at least min_imp greater than the confidence of any of its simplifications, where a simplification of a rule is formed by removing one or more conditions from its antecedent. This feature remedies the rule explosion problem resulting from the fact that in dense data-sets, the confidence of many rules can often be marginally improved upon in an overwhelming number of ways by adding conditions. For example, given the rule stating that A implies B with 99% confidence, there may be hundreds of rules of the form $A, X_1, X_2...X_n \rightarrow B$ with a confidence between 99% and 99.1% [39]. The improvement constraint trades this marginal benefit in confidence for a more precise rule set, with the added property that every returned rule consists entirely of items that are strong contributors to its predictive ability.

The algorithm presented enforces both the minimum improvement constraint and the minimum confidence constraint using a complicated set-enumeration search, in order to not break the completeness guarantee discussed by Srikant et al. in the previous sub-chapter. The set-enumeration search differs from regular association rule mining because candidate itemsets are not generated based on the (k-1) frequent itemsets, but based on the set-enumeration tree. More relevant is the constraint that the algorithm requires that a fixed consequent is specified on beforehand, and will thus only produce rules with a fixed consequent. Using the same argumentation as in the previous sub-chapter, this algorithm is because of this not directly applicable for the prediction problem at hand. However, the concept of a minimum improvement constraint for rules is novel, and is highly relevant to enforce as a post-processing technique for rule selection, and is reintroduced in Chapter 5.

4.3 A restricted rule mining approach

Based on the techniques discussed in the preceding sub-chapters no state of the art rule mining method, or algorithm, provides a solution that directly targets rules suited for predicting failures in oil production. However, the different techniques provides the necessary framework to build a solution which is probable to perform good at mining rules suited for predicting failures from event sequences. This sub-chapter will derive a complete method for mining association rules which adhere to the rule syntax and semantics defined in Chapter 3. This corresponds to incorporating the defined rule syntax and semantics into the rule mining algorithm, making it specialized for obtaining rules suited for prediction.

The suggested method is based on the MINEPI algorithm and minimal occurrences. MINEPI has proven

Algorithm 2: Discovery of restricted association rules

```
Input: An event sequence $\mathcal{S}$, a frequency threshold $min\_fr$, a maximum time $max\_time$
       and a reaction time $r\_time$
Output: All restricted rules with $lift > 1$
1.  $C_1 :=$ the set of episodes of size 1 in $\mathcal{E}_{\mathcal{S}}$;
2.  forall $P \in C_1$ do compute $mo(P) \mid i \in mo(P) <= max\_time$ ;
3.  $L_1 := \{P \in C_1 | P$ is frequent $\}$ ;
4.  $i := 1$;
5.  while $L_i \neq \emptyset$ do
6.      $C_{i+1} := \{P | P \in \mathcal{E}_{\mathcal{S}}$, all sub-episodes of $P$ are in $L_i\}$ ;
7.      $i := i + 1$;
8.      forall $P \in C_i$ do
9.          select sub-episodes $P_1$ and $P_2$ ;
10.         compute $mo(P)$ from $mo(P_1)$ and $mo(P_2) \mid i \in mo(P) <= max\_time$ ;
11.     end
12.     $L_i := \{P \in C_i | P$ is frequent $\}$;
13.     forall $P \in L_i$ do
14.         forall $E \in P$ do
15.             compute $mo(P_e) \mid i \in mo(P)$, $E$ occurs at the end time of $i$;
16.             compute sliding window count from $mo(P_e)$ ;
17.             compute $min\_time(P_e)$, $exp\_time(P_e)$, $max\_time(P_e)$ from $mo(P_e)$ ;
18.             compute $conf(P_e)$, $lift(P_e)$ ;
19.             $R_i := \{ P_e | P_e$ is frequent and $lift > 1$ and $exp\_time >= r\_time \}$;
20.         end
21.     end
22. end
23. forall $i$ and all $O \in R_i$ do output $O$ and $support(O)$, $conf(O)$, $lift(O)$, $min\_time(O)$,
    $exp\_time(O)$ and $max\_time(O)$ ;
```

to be an efficient algorithm for rule mining in event sequences, and the concept of minimal occurrences allows for easy integration of the constraints specified in the rule syntax. In addition, it provides a mechanism for easy computation of time information for rules. The method for incorporating item constraints described in Chapter 4.2.1 has been evaluated as unnecessary because the size of the actual datasets in ConocoPhillips is within feasible limits for a unconstrained search with modern computers, even with an extremely low support threshold. The mining approach chosen is broad, in the sense that no additional constraints are enforced during the discovery of frequent episodes. The only heuristic used to constrain the search space is the minimum support threshold. However, a maximum time specified by the user is enforced in the computation of the minimal occurrences, giving only episodes which has a upper time bound less than or equal to the maximum time defined. This reduces the amount of main memory needed in MINEPI to hold the minimal occurrences of episodes, but the savings will be moderate for dense datasets.

After the discovery of frequent episodes, in the rule generation phase, additional constraints are enforced for restricting the type of rules mined, and make the resulting rule set more compact:

- Rules are restricted to have a serial ordering between the antecedent and the consequent, but allowing for a parallel antecedent.

- Rules with a $lift$ less than 1 is discarded.
- Rules with an expected time of occurrence between the minimum reaction time of operators discarded.

These three constraints do not improve the performance of the frequent episode mining phase, but with the tight integration to the episode mining phase unwanted rules can be discarded efficiently. The suggested rule mining algorithm is outlined in Algorithm 2.

The suggested algorithm follows MINEPI, but with the following customizations: In line 2, only minimal occurrences with a time span less than max_time are considered. In line 9, the sub-episodes P_1 and P_2 are selected analogue to the parallel case in MINEPI, however it should be noted that for efficient computation of $mo(P)$, $|mo(P_1)| + |mo(P_2)|$ should be minimized. In line 10, $mo(P)$ is computed by a temporal join between $mo(P_1)$ and $mo(P_2)$ which enforce the max_time constraint. This allows for faster computation of $mo(P)$.

At line 13, the restricted rule generation phase starts. For each level of frequent episodes all rules are generated. Rules are generated from a frequent episode by sub setting the episode, line 14, such that all elements of the episode is considered as a single consequent, and rest of the elements of the episode make up the antecedent. Then at line 15, the minimal occurrences of the restricted episode P_e, $mo(P_e)$, is computed from $mo(P)$. $mo(P_e)$ is computed by selecting the minimal occurrences in $mo(P)$ where the consequent (E) occurs at the end of the time interval.

At line 16, the minimal occurrences of P_e are counted with respect to in how many sliding windows the rule would have been in if WINEPI was used. In general the minimal occurrences of P_e would suffice, because confidence can easily be computed from the minimal occurrences, but in order to compute lift for the rule, the probability of the episode occurring is needed. This probability is computed as the fraction of the sliding window count of the episode, and the total number of sliding windows over the sequence. The total number of sliding windows can easily be computed as: $((end\_time - star\_time)/step\_size) + (max\_time/step\_size) - 1$ where $start\_time$ and end_time is the start time and end time of the event sequence, respectively, and $step\_size$ is a selected step size for the imaginary sliding window. This step size is usually equal to the granularity of measurement in the data, e.g. one minute if sensors are measured each minute. If the restricted rule/episode P_e is frequent, and $lift > 1$, and the expected time of the rule is greater than the minimum reaction time of operators, the rule is added to the list of rules.

The suggested algorithm will output all rules which complies to the rule syntax and rule semantics defined in Chapter 3, and with a $lift$ value greater than 1, without the need of post-processing the mined rules to enforce the rule syntax.

4.4 Implementation concerns

In Chapter 3.4 an approach for enforcing the rule syntax, with respect to a serial ordering among events in the antecedent and the consequent, was outlined. This approach was motivated from the procedures available in SAS Enterprise Miner 5.2. The suggested approach in this chapter do not correspond to available procedures in SAS Enterprise Miner 5.2, hence a custom implementation or an extension of Enterprise Miner is needed for incorporating this procedure in the ConocoPhillips project.

The problem of discovering association rules for prediction can be handled within SAS Enterprise Miner

5.2 by using the approach outlined in Chapter 3.4, and enforcing the additional constraints in a post-processing phase. Such an approach has two downsides:

1. The data must be transformed to transactions before applying the sequence mining procedure. This transformation can be biased if the step size of the sliding window is large, or produce a vast number of transactions if the step size is small. If the size of the transformed transaction database is large, resource consumption can possibly render a full scale unconstrained mining process infeasible.

2. All combinations of rules, with no restrictions except maximum time and support threshold, is mined. Enforcing restrictions in a post-processing phase for this amount of rules will require multiple passes over the original database, hence be less efficient than enforcing the constraints as an integrated process of the rule mining algorithm using minimal occurrences.

However, performance issues are of little concern in the ConocoPhillips case because the rule mining is done off-line. No studies have been done in this thesis of the feasibility, with respect to running times and hardware requirements, of such a solution in SAS Enterprise Miner 5.2. A solution built on the existing rule mining procedures in SAS Enterprise Miner 5.2 will allow for fast implementation in ConocoPhillips. Because of this it is interesting, using a pragmatic view, to pursue such a solution despite the theoretical downsides. How to enforce the constraints which are enforced in the suggested rule mining algorithm during a post-processing phase in Enterprise Miner 5.2 is revisited in Chapter 5.4.

Chapter 5

Rule selection

Rule selection is the process of identifying and selecting rules which are suited for prediction from a large rule set. Given the vast number of rules returned from the rule mining algorithms a process is needed for selecting the rules which are truly interesting and well suited for prediction. Considering the specific case of predicting a single failure event, it is clear that not all rules found by the rule mining algorithms will influence the target variable. Rules will also be of different quality with respect to the information the rule gives about the association between events enclosed in the rule. These factors motivate for a process where redundant rules are removed, and rules are ranked according to their interestingness, before a subset is submitted to a domain expert for validation. This chapter will introduce the theoretical field of interestingness, and derive a suggested approach for rule selection based on the purpose of predicting single events.

5.1 Interestingness

Interestingness is a key issue in this thesis, and is the foundation for identifying interesting rules, and perhaps removing non-interesting rules. In order to automatically rank rules based on a notion of interestingness, an interestingness measure is used. How to define interestingness for rules is a difficult task, and highly domain specific. Interestingness also depends on the purpose of the rules, and can be highly subjective. Hence a great variety of measures have been developed, and been the basis for extensive research.

The literature distinguishes between two kinds of measures: subjective (user-oriented) and objective (data-oriented). Subjective measures take into account the user's goals and domain knowledge [27], whereas only the data cardinalities appear in the calculation of objective measures [6]. These objective measures are defined in terms of the frequency counts tabulated in a 2 x 2 contingency table, as shown in Table 5.1. Objective measures involve analysing the rule's structure, predictive performance, and statistical significance. In association rule mining, such measures include support and confidence [27]. In the prediction case, is users subjective opinion of a rule less relevant because the rules is to be included in a reasoning structure for prediction. Hence it is the rule strength, predictive performance, information content and statistical significance which is important. This thesis will because of this only consider objective measures. However, when "strong" rules are found, the subjective interestingness is evaluated by a domain expert, ensuring the actual interestingness and validity of the rules.

	B	$\overline{B}$	
A	f_{11}	f_{10}	f_{1+}
$\overline{A}$	f_{01}	f_{00}	f_{0+}
	f_{+1}	f_{+0}	N

(a) Frequency counts

A/B	1	0	
1	$P(AB)$	$P(A\overline{B})$	$P(A)$
0	$P(\overline{A}B)$	$P(\overline{AB})$	$P(\overline{A})$
	$P(B)$	$P(\overline{B})$	1

(b) Joint distribution

Table 5.1: 2 x 2 contingency table for events A and B

There exist various evaluations of objective interestingness measures, with different approaches to interestingness. Good evaluations of interestingness measures can be found in [44], [16], [43] and [22]. The goal of these evaluations is not to conclude with a single measure, but to describe the various key properties of a measure. Due to different purposes and domain specific concerns, the conclusion of which interestingness measure to choose is highly domain specific. A relatively recent approach to rule interestingness is to develop measures based on information theory. Hence there exist two groups of objective interestingness measures, pure probabilistic ones and information theoretic ones. This chapter will introduce the most common probabilistic and information theoretic interestingness measures in the literature, and evaluate them with respect to predicting failures in oil production. At last, a recommendation for use in ConocoPhillips is given.

5.1.1 Desired properties of an objective measure

Although there are numerous measures available for evaluating association patterns, a significant number of them provide conflicting information about the interestingness of a pattern [44]. This has provoked a significant amount of research towards establishing a set of global properties an interestingness measure should possess. Piatetsky-Shapiro [37] have proposed three key properties a good measure M should satisfy:

P1: $M = 0$ if A and B are statistically independent.

P2: M monotonically increases with $P(AB)$ when $P(A)$ and $P(B)$ remain the same.

P3: M monotonically decreases with $P(A)$ (or $P(B)$) when the rest of the parameters ($P(AB)$ and $P(B)$ or $P(A)$) remain unchanged.

To see the rationale of these properties consider the rule $A \rightarrow B$ and Table 5.2.

Property	**P(A)**	**P(B)**	**P(AB)**	**Measure**	**Rationale**
P1	$P(AB) = P(A) * P(B)$			0	Independence (no information)
P2	*constant*	*constant*	*increases*	should increase	Belief in co-occurrence increase
P3(1)	*increases*	*constant*	*constant*	should decrease	Belief in co-occurrence decrease
P3(2)	*constant*	*increases*	*constant*	should decrease	Belief in co-occurrence decrease

Table 5.2: Rationale behind the Piatetsky-Shapiro properties

If the events A and B are statistically independent (the case of P1), then clearly this rule should be omitted, or scored low, because knowing A will not inflict our belief about B. In other words the rule

$A \to B$ gives no new information. In the case of P2, when $P(A)$ and $P(B)$ remain the same and $P(AB)$ increases, then our belief about the co-occurrence of A and B becomes stronger, and this should be reflected by the interestingness measure in order to properly distinguish rules. Property 3 describes the case where $P(AB)$ and $P(B)$ remain fixed, and $P(A)$ increases. In this case the situation becomes slightly more difficult to grasp. The belief in A alone is increasing, but our belief in the co-occurrence of A and B remain the same. If A is more likely to happen alone, then our actual belief in the co-occurrence is less, because there must exist more counterexamples of the rule (situations where A is true, but B is false), hence our confidence in the correctness of the rule is lower.

Lallich et al. [43] argues for the importance of counter-examples. The example of a rule $A \to B$ is defined as if A occurs in a transaction then B also occur in the same transaction. Then a counter-example becomes the situation where A occurs in a transaction, but B does not. Lallich et al. introduce two properties an interestingness measure suited for association rule mining should incorporate, based on the principle of counter-examples [43]:

P4: A measure must permit a clear choice between $A \to B$ and $A \to \overline{B}$, since the examples of one are the counter-examples of the other.

P5: Asymmetric measures which respect the nature of transactional rules are preferred.

Property four stresses the fact that an interestingness measure needs to give different values for positive and negative rules, because a reasonable interestingness measure for association rule mining need to consider counter examples. If the rule $A \to B$ has a large number of examples versus counter-examples then the rule $A \to \overline{B}$ must have a low number of examples versus counter-examples. A rule with a small number of examples and a large number of counter-examples is clearly not interesting, because we have little belief in the rule. This example justifies that an interestingness measure should distinguish between the positive and the negative rule. Property five states that symmetric measures pose a problem for association rules. A symmetric measure gives the same value to the rule $A \to B$ and $B \to A$. This is a problem because while the rules $A \to B$ and $B \to A$ have the same examples, they do not have the same counter-examples [43]. The differences in number of counter-examples of the rules affect our belief in each of the rules. Hence an appropriate interestingness measure for association rules should distinguish between $A \to B$ and $B \to A$, in other words be asymmetric.

5.1.2 Probabilistic measures

Probabilistic measures for interestingness have been used to evaluate interestingness of rules since the introduction of rule mining. Based on the frequency of occurrence in publications, the most common interestingness measures have been selected and are presented in Table 5.3. These include the support [1], confidence [1], lift [9], odds ratio [35] and Piatetsky-Shapiro [37] measures. Other popular measures not evaluated in this thesis, because they are less suited for the domain at hand, are Gini-index [8], ϕ-coefficient (a variant of χ^2 not dependant on the size of the database) [4] and Conviction [9].

Support and confidence
Support and confidence were introduced in Chapter 2.1. Following the notation of this chapter, and Table 5.1, support of the rule $A \to B$ is defined as

Measure	**Formula**	**P1**	**P2**	**P3**	**P4**	**P5**
Support	$P(AB)$	No	Yes	No	NA	No
Confidence	$\frac{P(AB)}{P(A)}$	No	Yes	No	Yes	Yes
Lift	$\frac{P(AB)}{P(A)P(B)}$	Yes	Yes	Yes	Yes	No
Odds ratio	$\frac{P(AB)P(\overline{AB})}{P(A\overline{B})P(\overline{A}B)}$	Yes*	Yes	Yes	No	No
Piatetsky-Shapiro	$P(A,B) - P(A)P(B)$	Yes	Yes	Yes	Yes	No

* when normalised

Table 5.3: Usual probabilistic measures of interest

$$supp(A \rightarrow B) = P(AB) = \frac{f_{11}}{N} = \frac{\#(A \text{ and } B)}{\#transactions} \tag{5.1}$$

and the confidence of $A \rightarrow B$ is defined as

$$conf(A \rightarrow B) = P(B|A) = \frac{P(AB)}{P(A)} = \frac{supp(A \rightarrow B)}{supp(A)} \tag{5.2}$$

Support is the fraction of numbers of co-occurrences of events A and B, and the total number of possible occurrences. The rationale for using support is that mined rules should satisfy a minimum boundary, in order to be regarded as significant enough for a rule to be considered valid. Confidence may be interpreted as the conditional probability of B given that A has occurred, or how often we could expect B to happen if we know that A has happened. This conditional property is desired and important for predicting events, because if we observe the antecedent, the confidence represents the belief we now have in the consequent happening. Conditional probabilities can also be combined in chain of rules, making it possible to state a belief of an end event from a long chain of rules, given that some events in the chain have occurred.

The anti monotonicity principle of support gives a desired heuristic to efficiently search through all combinations of possible rules. This property comes from two facts: Any subset of frequent itemsets is frequent, and any superset of a non-frequent itemset is non-frequent. Hence all non-frequent itemsets, and their supersets, encountered in the search process may be excluded in the following iterations. This leads to a dramatic increase in efficiency of the A-priori algorithm, and its descendants.

However, the usefulness of support and confidence as a measure of interestingness is questionable. First, algorithms based on support pruning generate a large number of rules, many of them of little interest. Moreover, the support condition, at the core of the extraction process, neglects rules with a small support though some may have a high confidence, thus being genuinely interesting [43]. Lowering the support threshold to remedy this problem makes the algorithms sensitive to noise, and dramatically increases running times and the number of rules mined. Second, the support and confidence conditions alone do not ensure rules with real interest. In the case of property 1 - Independence ($P(B|A) = P(B)$) the rule gives no new information, but may still have a high value of confidence (e.g. $P(a) = 0.8, P(b) =$

$0.9, P(ab) = 0,72, P(b|a) = 0.9$). This implies that a rule can have high confidence, but will not give any increased belief in the consequent happening, even if the antecedent has occurred. Confidence and support thus fails to satisfy property 1 and 2 of the Piatetsky-Shapiro requirements. Hashler and Hornik [17] also argues that confidence fail to take the probabilistic properties of the mined data into account. They prove that confidence increase with the consequent of the rule, hence biasing rules with high support of the consequent. It should be clear from this discussion that other measures than support and confidence must be examined in order to filter out the truly interesting rules.

Lift
Another measure is *lift* (first introduced as *interest* by Brin, et al. [9]) which is defined as

$$lift(A \rightarrow B) = \frac{P(AB)}{P(A)P(B)} = \frac{conf(A \rightarrow B)}{supp(B)} \tag{5.3}$$

A lift value greater than 1 indicates a positive correlation. Lift measures the increase in probability of the consequent given the antecedent (episodes containing A tend to contain B more often than episodes that do not contain A), while a lift value smaller than 1 indicates a negative correlation between the events. If the lift value is (near) 1 then the events appear together as often as expected due to random chance. Lift has a clear, concrete meaning for the user. A lift value of 2 means that the number of examples of the rule $A \rightarrow B$ is twice what is expected under independence. Hence, a customer who buys A is twice as likely to buy B than the general consumer. Similarly, because lift is symmetric, and that the examples of $A \rightarrow B$ are also those of $B \rightarrow A$, he who buys B is also twice as likely to buy A. Lift thus fails to satisfy property P5. However, lift satisfy property P1(if normalised), P2 and P3 of the Piatetsky-Shapiro requirements.

Considering the case of predicting events, lift provides an important property. Rules with a lift value of 1 are unwanted because they provide no new information. In theory both negative correlations and positive correlations, $lift < 1$ and $lift > 1$ respectively, can be interesting. Negative correlations provide the information that by observing A the belief in B is less than by chance, and vice versa for positive correlations. In the context of giving warnings about possible failure events we want to obtain rules which indicate that by observing some event, or combination of events, the belief that the failure event is going to happen is increased compared to the target happening by chance. By restricting rules to have $lift > 1$, as done in Chapter 4, one can ensure that only rules that describes a positive correlation is mined. Lift also provides important information for prediction, because it describes a true influence of the rule in our belief in the consequent. Thus, where confidence fails to give true information about the increase of belief in the consequent, lift provides this important property for ensuring interesting rules.

Hashler and Hornik [17] argues that lift is biased towards infrequent rules. The lift measure produces higher values for very infrequent items; i.e. the highest lift values occur close to the boundary of the selected minimum support. This becomes a problem if the minimum support threshold is altered, which will result in a completely altered ranking of rules. With a low support boundary this fact also makes the lift measure sensitive to noise, because noise exceeding the support boundary may be assigned high lift values. The fact that lift is symmetric also poses a problem, as stated in P5, and makes it less appropriate as the only measure of interestingness for association rules.

Odds ratio
Odds ratio originates from the classic work by Mosteller [35] on association analysis of contingency tables, and is defined as:

$$oddsratio(A \rightarrow B) = \frac{P(AB)P(\overline{AB})}{P(A\overline{B})P(\overline{A}B)} \tag{5.4}$$

Odds ratio represents the odds for obtaining the different outcomes of a variable. For example, consider the frequencies given in Table 5.1. If B is present, then the odds of finding A in the same transaction is f_{11}/f_{01}. On the other hand, if B is absent, then the odds for finding A is f_{10}/f_{00}. If there is no association between A and B, then the odds for finding A in a transaction should remain the same, regardless of whether B is present in the transaction. Odds ratio use these odds, $(f_{11}f_{00}/f_{01}f_{10})$, to determine the degree to which A and B are associated with each other [44].

Odds ratio satisfy all three desired properties defined by Piatetsky-Shapiro for an interestingness measure, but lacks a clear formalism for describing interestingness in association rules. In addition, the odds ratio formula includes all combination of probabilities, hence it will not differentiate between the rule $A \rightarrow B$ and $B \rightarrow A$, and the rules $A \rightarrow B$ and $A \rightarrow \overline{B}$. Hence, odds ratio fails to satisfy P4 and P5. In practice this implies that a high odds ratio ensures that the events A and B are associated, but the rule may prove to be truly uninteresting.

Piatetsky-Shapiro
Piatetsky-Shapiro presented an interestingness measure in [37] satisfying the desired properties presented in the same article. It is defined as:

$$PS(A \rightarrow B) = P(A, B) - P(A)P(B) \tag{5.5}$$

This measure measures the distance from independence, hence measuring the same distance as $lift$, but with different properties. With regard to P4 and P5 the Piatetsky-Shapiro(PS) measure satisfy P4, but do not satisfy P5. The PS measure also lacks the clean formalism for easy interpretation of the meaning of the measure. Because it is difficult to interpret, it is less suited for presentation to users, and as a measure stating the belief in consequent events occurring.

5.1.3 Information theoretic measures

Measures based on information theory have been developed, and are extensively used in various classification tasks, and decision trees in specific. Such measures have also been used in the literature as a measure of interestingness for association rules. However, there has been little work done on developing specific interestingness measures based on information theory suited for association rules. Among the objective measures of rule interestingness, the information-theoretic measures are particular intelligible and useful since their interpretation of rules in terms of information often corresponds to rules suited for prediction. Recall the basics of information theory introduced in Chapter 2.2. The idea of using information theory to measure interestingness is motivated by the assumption that the amount of information contained in a rule is a key differentiator between rules. In the task of predicting failures one can argue that those rules which provide the most information about the failure, will perform best at predicting the failure event. Given that only a subset of all rules will be available for prediction, one will choose those rules maximizing the local information about dependencies between two events, with the goal of maximizing the encoded information about the failure event described by the set of rules.

The information theoretic measures commonly used to evaluate rule interestingness are the Shannon conditional entropy [40], the average mutual information [12] and the J-measure [41]. These measures are shown in Table 5.4.

Measure	**Formula**	**P1**	**P2**	**P3**	**P4**	**P5**
Conditional entropy (H_c)	$-P(B\|A)log_2P(B\|A) - P(\overline{B}\|A)log_2P(\overline{B}\|A)$	No	No	No	No	No
Average Mutual Information (MI)	$P(AB)log_2\frac{P(AB)}{P(A)P(B)} + P(A\overline{B})log_2\frac{P(A\overline{B})}{P(A)P(\overline{B})} + P(\overline{A}B)log_2\frac{P(\overline{A}B)}{P(\overline{A})P(B)} + P(\overline{AB})log_2\frac{P(\overline{AB})}{P(\overline{A})P(\overline{B})}$	Yes	Yes	Yes	No	No
J-measure (J)	$P(A)[P(B\|A)log\left(\frac{P(B\|A)}{P(B)}\right) + (1-P(B\|A))log\left(\frac{(1-P(B\|A))}{(1-P(B))}\right)]$	Yes	No	No	Yes	Yes

Table 5.4: Usual information theoretic measures of interest

Conditional entropy
The Shannon conditional entropy [40] is a measure directly derived from information theory and easily applied to rules. It measures the average amount of information of the consequent given that the antecedent is true [7]. The Shannon conditional entropy is given as:

$$H_c(A \to B) = -P(B|A)log_2P(B|A) - P(\overline{B}|A)log_2P(\overline{B}|A) \tag{5.6}$$

The Shannon conditional entropy is less appropriate for association rules because it fails to satisfy P4, because it does not distinguish between $A \to B$ and $A \to \overline{B}$. In addition it is symmetric, hence fails to satisfy P5. It also does not incorporate a notion of increase in belief about the consequent, given the antecedent, hence it is difficult to interpret by an user.

Mutual Information
The average mutual information [12] is at the core of information theory and considers the full joint distribution of the antecedent and the consequent. The average mutual information is defined as:

$$\begin{aligned} MI(A \to B) = P(AB)log_2\frac{P(AB)}{P(A)P(B)} + P(A\overline{B})log_2\frac{P(A\overline{B})}{P(A)P(\overline{B})} + \\ P(\overline{A}B)log_2\frac{P(\overline{A}B)}{P(\overline{A})P(B)} + P(\overline{AB})log_2\frac{P(\overline{AB})}{P(\overline{A})P(\overline{B})} \end{aligned} \tag{5.7}$$

Mutual information is an entropy-based measure for evaluating the dependencies between variables. It represents the amount of reduction in the entropy of a variable, when the value of a second variable is known. If the two variables are strongly associated, then the amount of reduction in entropy, i.e. its mutual information, is high [44]. For example, if A and B are independent, then knowing A does not give any information about B, and vice versa, so their mutual information is zero. The average mutual information corresponds to the desired metric wanted for prediction, obtaining rules that give the most information about the consequent. However, the mutual information is always non-negative and is symmetric. Because it includes the full joint distribution of A and B it also fails to distinguish between $A \to B$ and $A \to \overline{B}$. Hence it does not satisfy P4 and P5.

J-measure
Another interestingness measure widely discussed in the literature is *J-measure* [41], which measures the average amount of information contained in a rule, and can be seen as the directed mutual information. The amount of information contained in a rule can be used to measure how important the rule is. Because of this is J-measure the most commonly used interestingness measure in the literature. Average J-measure for a rule $A \rightarrow B$ is defined in [41] as:

$$J(B : A = a) = P(A) \left[P(B|A) \cdot log \left(\frac{P(B|A)}{p(B)} \right) + (1 - P(B|A)) log \left(\frac{(1 - P(B|A))}{(1 - P(B))} \right) \right] \tag{5.8}$$

J-measure describes the average mutual information between events A and B, with the expectation taken with respect to the a-posteriori probability distribution of B. The J-measure combines a bias toward more frequently occurring rules (the first term $P(A)$), with the degree of surprise in going from a prior probability $P(B)$ to posterior probability $P(B|A)$ [21].

Considering the application of J-measure in prediction, it does not provide a metric which can be easily interpreted by a user. Where confidence states the amount of belief we have in the consequent of a rule given that the antecedent occurred, and lift states the increase in belief, J-measure cannot be interpreted in an analogue way. However, as a measure for selecting the rules that will give a good predictive ability, it may be better suited than other interestingness measures. In general J-measure is not symmetric, but it increases at both sides of independence (zero at independence). This is intuitively correct because also strongly negatively correlated rules give valuable information. As discussed in the case of lift, negatively correlated rules can be discarded for a pure prediction purpose.

By requiring that rules should have a $lift > 1$, J-measure get the wanted properties of an interestingness measure for association rules. Compared to confidence and lift, high values of J-measure for a rule tend to correspond to rules with both high confidence and lift. Similarly, rules with a low J-measure tend to have a low confidence and a low lift. Evaluating Equation 5.8 confirms this by realising that both lift and confidence is a part of the equation. In addition the body of the equation is multiplied with the probability of the antecedent to obtain the average J-measure. This implies that average J-measure is biased towards rules with a frequently occurring antecedent. This property may prove well suited for prediction, because it ensures that the top ranked rules will be rules that actually have a decent probability of occurring, in opposite to lift which gives the highest values to rules with a low probability of occurring.

5.2 Selecting rules for prediction

As introduced in Chapter 4.2 the preferred rule mining approach is to do unfocused rule mining. The result of this is a possibly large number of rules, which never will influence the target event. However, if the only purpose of the rules is to be used for prediction of a single event, it is possible to select only those rules which can influence the target event. It is two aspects of rules that can be discarded because they do not influence the prediction of the target event:

1. Rules which are redundant specifications of more general rules with the same predictive strength.
2. Rules which when combined with other rules, forming a chain of rules, never will lead to the target event.

Redundant specifications
In Chapter 4.2.2 the concept of minimum improvement was introduced as an integrated part of a constraint based rule mining algorithm. This algorithm was discarded due to the restriction that only rules with a fixed consequent could be mined, but the concept of minimum improvement is novel and highly relevant as a post-processing technique. Recall that the minimum improvement constraint required that a specification of a rule must have a confidence larger than the original rule. E.g. that the rule $A, B \rightarrow C$ is only interesting, with respect to prediction, if it has a higher confidence than the more general and more applicable rule $A \rightarrow C$. It is often the case, that by adding an element to the antecedent, the resulting rule will only have a marginal difference in confidence. For the purpose of prediction, such specifications are deemed to be uninteresting, because they will not give any new information about the consequent, and the more general and more applicable rule can be used instead.

However, also negative improvement is considered as interesting in this thesis, because negative improvement can be important with respect to the soundness of the rule set. Consider the example where the rule $A \rightarrow C$ has a confidence of 90%, and the more specific rule $A, B \rightarrow C$ has a confidence of 40%. This is an important rule, because if we observe A we have 90% belief in that C is going to happen. If we now observe B, and the rule $A, B \rightarrow C$ has been removed, we still have 90% belief in C happening, in opposition to the 40% belief we would have had if the rule $A, B \rightarrow C$ had been present. It should be clear that, if the statement of current belief in C should be sound, specifications of rules should be included even when they have a negative improvement. This issue is further discussed in Chapter 6.

Combining rules
In Chapter 4.2 a focused rule mining approach using item constraints was outlined, but it was argued that the performance gain will be neglectible, or perhaps even worse, than doing unfocused rule mining. However, the idea of selecting only those rules that, when combined, can be used to predict a target event still applies as a post-processing technique for reducing the number of uninteresting rules.

Various approaches for combining rules are possible, but are not the focus of this thesis. Combining rules for reasoning is the purpose of the related work done by Helland [20]. However, as a technique for removing uninteresting rules some of the elements discussed in Helland's work are important in the post-processing of a large rule set. One important element is how to build a graph from the rule set, ending in the failure event. There exists an algorithm for efficient building of a hyper graph from association rules, named Association Rule Network (ARN) [10]. The ARN algorithm takes a set of association rules and a goal node as input, and level wise builds a hyper graph outwards from the goal node. At each level the rules which have a consequent at the actual level are included. This is done recursively until either a predefined number of levels are reached, or there are no more rules to add. The rules not participating in the graph can be discarded, since they never will influence the target event, resulting in a greatly reduced rule set. For an illustration of the concept, see Figure 5.1.

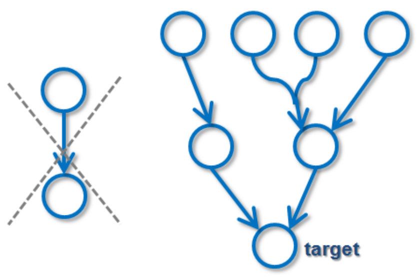


Figure 5.1: Example illustrating removal of rules which do not lead to the target

5.3 Suggested rule selection approach

Based on the material presented in this chapter, and the rule mining algorithm derived in Chapter 4.3, this thesis suggests a four step process for post-processing a large rule set. The ultimate goal is to obtain a smaller set of rules, which are truly interesting with respect to prediction of failure events.

1. Removing specification rules with a low improvement
2. Removing rules that, when combined, not influence the target event
3. Applying an interestingness measure to rank the rules
4. Submitting the rules to an expert for validation

Selecting a specific interestingness measure to rank the rules is a difficult task. From the various measures presented in this chapter it should be clear that all measures have some deficiencies, and that no single measure hold all properties wanted for a interestingness measure suited for evaluating rules for prediction. The focus in the literature has been to develop one general measure that exhibits all the wanted properties for association rules, but most of them have fail to do so because the concept of interestingness is highly domain specific. This thesis takes the stand that instead of developing one single measure, various measures can be combined to solve the task at hand. From the discussion of the various measures in this chapter it is clear that confidence and lift holds important properties for the problem at hand, but both alone will not suffice. Hence, a combination of confidence and lift will probably be better for identifying interesting rules for the task at hand. However, how to sort rules based on both measures is unclear.

From the evaluation of the various information theoretic measures, J-measure combines lift and confidence in an appropriate way, but has a bias towards frequent occurring rules. This bias is appropriate to distinguish rules with relatively equal support and lift, because a rule occurring more frequently will more often be applicable and hence more relevant in predicting the failure event. However, if a rule is very frequent this bias can dominate confidence and lift. J-measure is also positive for negatively correlated rules, but this can be easily overcome by combining lift and J-measure, requiring that rules should have a lift value greater than one. By selecting the rules with largest J-measure the amount of information about the associated events in the selected rule set will be maximized. It is shown in other domains, e.g. classification, that maximizing this information yields better predictive results than other selection criteria [38].

One good way to obtain strong rules with good predictive ability can be to combine J-measure, lift and confidence in the selection of rules. Because J-measure combines lift and confidence it can be used to do an initial selection of a subset of rules that can be presented to a domain expert. The expert should validate the rules, ensuring that rules are truly interesting, and that they hold in the real world. The expert should also check that the bias of J-measure not dominates confidence and lift. This can only be done if the expert is shown all three measures for a rule, and the probability of the antecedent occurring. Based on this information, the expert can use his own experience to evaluate the bias of the different interestingness measures, while validating and selecting the correct rules to include in a reasoning structure for prediction. These rules are then used as basis for real-time prediction of oil production problems.

The suggested use of an expert is motivated by the realization that rules mined based solely on data cannot be guaranteed to represent actual causal relationships that hold in the real world. It is important to realise that before the expert is given a set of rules this rule set must have been automatically pruned, e.g. by using the techniques of this chapter, in order for the validation task to be feasible.

Preliminary tests performed on the rule set mined from the ConocoPhillips data, which consists of approximately 443 000 rules, indicates that the second removal process, where rules are combined, decreases the number of available rules to approximately 143 000 rules. See Appendix A for implementation details. Recall from Chapter 3 that confidence and lift is a part of the defined rule syntax, and that computation of J-measure can be done fast from confidence, lift and support, hence the approach is feasible in practice.

5.4 Implementation concerns

Considering an implementation using SAS Enterprise Miner 5.2, all the above steps can be custom implemented inside SAS Enterprise Miner 5.2, which allows for easy integration in ConocoPhillips. However, as discussed in Chapter 4.4, standard rule mining procedures in SAS Enterprise Miner 5.2 can be used to obtain regular sequence rules, and the defined rule syntax and semantics can be enforced in a post-processing phase instead of doing it as an integrated part of the rule mining algorithm. This sub-chapter will outline an implementation of the suggested rule selection process in Enterprise Miner 5.2.

How the defined rule syntax and semantics are enforced during post-processing in Enterprise Miner 5.2, is described in Chapter 3.4 and will not be handled here. The customized rule mining approach suggested in Chapter 4.3 enforces an additional constraint on the rules besides the rule syntax and rule semantics. Only rules with a $lift > 1$ was generated. This constraint can easily be enforced in SAS Enterprise Miner 5.2, because the $lift$ value is available for all rules, and a simple selection can be done.

Step 1 of the process, removing specification rules with a low improvement, can be done in the following way: For each event, select all rules with the event in question as the consequent. Subset the resulting rule set according to the length of the antecedent. For all rules in each subset k, check if their confidence is at least min_imp greater than all rules sharing elements in the antecedent from the $k-1$ subset.

Step 2 of the process, removing rules that do not influence the target event, can be done in the following way: Select all rules which has the target event as consequent. For each level, build a list of those events that are contained in the antecedent of the resulting rules. Select those rules that have one of the events from the list as consequent, and which has not been selected before. Generate new consequent list. Do this until a predefined number of levels have been reached.

Step 3 of the process, ranking rules by J-measure, can be done by computing J-measure from $support$, $confidence$ and $lift$ of the rule. J-measure can be written in terms of $support$, $confidence$ and $lift$ as:

$$J(A \rightarrow B) = \frac{supp}{conf} * \left[conf * log_2\,(lift) + (1 - conf) * log_2 \left(\frac{(1 - conf)}{\left(1 - \left(\frac{conf}{lift}\right)\right)} \right) \right] \tag{5.9}$$

where $supp = support(A \rightarrow B)$, $conf = confidence(A \rightarrow B)$ and $lift = lift(A \rightarrow B)$.

Chapter 6

Discussion

This thesis has tried to develop a theoretic framework for discovering interesting rules from event sequences with the purpose of pre-warning oil production problems. The result is two folded; a clearly defined rule syntax and rule semantics for targeting rules suited for prediction, and a two step process for discovering interesting rules from event sequences. The two step process consists of two phases derived in Chapter 4 and Chapter 5 respectively:

- Restricted association rule mining
- Rule selection

The two step process is general in the sense that techniques for targeting rules suited for prediction is defined, but how these techniques are distributed in the two step process is of less importance. The suggested solution comes with a recommendation of where to apply the techniques to achieve efficiency improvements, but is not mandatory. In general these techniques can be seen as a way of enforcing constraints on the rule mining process. All constraints enforced in the restricted rule mining algorithm in Chapter 4.3, are possible to enforce in the rule selection phase. However, the constraints enforced in the rule selection phase are fixed, because they do not easily integrate into rule mining algorithms. This generalization opens up for a different implementation, using other rule mining algorithms than the one derived in Chapter 4.3, and enforcing the remaining constraints in the rule selection phase.

While deriving the framework, some key issues were encountered and many assumptions and choices were made. First, this chapter will discuss these assumptions. Second, two possible solutions for a complete process following the framework are outlined. Third, a discussion of the validity of the proposed method will follow. Next, an evaluation of the overall process of going from raw data, via association rules, to pre-warning of process problems is done. Finally, an explanation of how this framework can be applied in the ConocoPhillips case is given.

6.1 Assumptions

In Chapter 3 a rule syntax and semantics for rules suited for prediction were defined. As indicated in the problem description, rules which should be used for the purpose of prediction differ from regular association rules. This statement is not an absolute truth, and shall not be regarded as such. In general it is possible to use standard association rules to do prediction. To turn it around, this thesis is founded on the understanding that specific solutions tends to perform better for the task at hand than highly general ones. Then it becomes logical to pursue a definition of rule types that can restrict the rule mining approach, in a way that makes it more likely to perform better at prediction than a general approach.

With this in mind, a set of assumptions and choices were made in order to define a restricted rule syntax and semantics for rules suited for prediction. These assumptions and choices were:

1. Rules with one element in the consequent is better suited for prediction of a single event
2. Rules pointing forward in time is better suited for prediction of time based events
3. Interesting relationships are found by looking at combinations of events, not only single events
4. The time between occurrences of events in the antecedent do not influence the process system
5. The ordering of events in the antecedent does not give different process behaviour
6. Rules which incorporate time information are better suited for prediction
7. A defined maximum time of validity for a rule is essential for prediction
8. Rules with an expected time of influence below operators reaction time can be discarded

All assumptions and choices have been discussed in Chapter 3, but some have a broader implication which calls for an extended discussion.

Assumption 1: The choice of restricting rules to a single consequent is one of the most far reaching choices because it does not correspond with the use of regular association discovery algorithms. Regular association discovery algorithms produce rules with possible multiple events in the antecedent and the consequent. However, the argumentation for making this assumption is strong. A rule on the form $A \rightarrow B, C$ will not give any new information with respect to prediction of a single failure event, compared to the two single events $A \rightarrow B$ and $A \rightarrow C$, that also must exist. In addition, as noted in Chapter 3, reasoning becomes difficult with a multiple consequent. Hence, should such rules be discarded.

Assumption 2: The assumption that only rules pointing forward in time are useful in prediction of time based events, has a possible consequence dependant on the way it is enforced in the rule mining phase. If this assumption is enforced as a constraint during candidate generation, the completeness of the rule mining algorithm is broken, analogue to the case of item constraints discussed in Chapter 4.2.1. This thesis has chosen a different approach, to enforce this constraint during rule generation, keeping the rule mining algorithm complete.

Assumption 3: The choice of using a multiple antecedent for rules is a choice contradicting the idea of restricting the rule space as much as possible, because it will result in a dramatically increased number of possible rules. However, this choice is essential, because it is reasonable to believe that with the relatively small amount of events defined in the process system, it is probable that the operators already would have discovered these associations by experience. In such a case, the prediction system will give little value. However, if the number of elements in the antecedent is increased to two events, the increase in possible combinations of events in the antecedent will make the amount of possible rules larger than the operators are probable to learn by experience. Hence, this thesis is founded on the assumption that interesting associations, previously unknown to the operators, can be found by combining events in the antecedent.

Assumption 5: The assumption that ordering among events do not inflict the behaviour of the process system is a far reaching assumption. This assumption implies that a parallel antecedent is sufficient, hence discards the direct application of sequence rule mining algorithms. This assumption is highly domain specific, and is based on discussion with the domain experts in ConocoPhillips. As noted in Chapter 3, the conclusion is that it should suffice to consider a parallel antecedent, even if a serial antecedent can provide important information. The logic behind this argument is that a minimal amount of predictive ability is lost by considering a parallel antecedent. Mining rules using a parallel antecedent will capture the same associations between the events in the antecedent and the consequent, but may result in false warnings or underestimation of the importance of specific ordering of events.

In addition, the choice is motivated from a reasoning perspective; the work done by Helland [20] only considers a parallel antecedent. However, using a pragmatic view it is apparent that an approach using a serial antecedent has the ability to capture more information. However, assuming a serial antecedent when the ordering actually does not matter in the real world, can also result in an underestimation of the co-occurrence of specific events. It will also result in a larger number of possible rules. The only way to conclude on this issue in a satisfying way is to try both approaches, with a reasoning structure that handles both serial and parallel antecedent, and measure their respective predictive performance on the actual dataset. This is beyond the scope of this thesis, and the initial assumption is regarded as the most probable, based on the domain experts conclusions.

Assumption 6: The last choices made in Chapter 3 was with respect to time. The statement about a maximum time of validity for rules is an absolute necessity, but can result in difficulty in practical applications. In a system for predicting failures in a process system, it is difficult to set a maximum time of influence between events. For example, it may be that events happening days in advance actually, in the last instance, results in a failure. Such associations are clearly important to capture, but where should the time limit for influence between events be set? If the time limit of influence is set too large, the number of possible associations discovered increases, and the rule mining process becomes more sensitive to noise. There exists no absolute conclusion to this problem, and it is advised to derive the definition of maximum time validity for rules in close cooperation with domain experts.

Assumption 8: If the time information (assumption 7) and the minimum time boundary is discarded, this greatly reduce the complexity of the rule mining algorithm. Standard algorithms can be used instead of MINEPI in the restricted rule mining approach, because it is not necessary to store occurrences of episodes in the algorithm. The resulting rule semantics will then be more similar to the WINEPI semantics. The consequence of not enforcing the minimum reaction time constraint is a larger set of mined rules. In addition, will rules with a low expected time between the antecedent and the consequent be included in the prediction system in a similar way as other more helpful rules. The question of whether to enforce the last two choices of the time semantics and time syntax becomes an implementation concern. The properties enforced by these constraints are not essential with respect to prediction, but can reduce the number of rules mined. It should be noted that enforcing these time constraints in MINEPI is trivial. Hence, the suggested conclusion is to enforce this part of the syntax only if MINEPI is the preferred choice of rule mining algorithm.

6.2 Suggested solutions

The assumptions and choices discussed affect the number of possible rule mining methods available. Discarding the suggested rule syntax and semantics implies in practice that standard association rule mining algorithms can be applied. After transforming the event sequence to a transaction database, general rules can be mined without any complex customizations. This shows that there exists numerous easy applicable solutions to the rule mining problem in this thesis, and from a theoretically point of view the problem of mining rules from event sequences is trivial. The consequence of solving the problem in such a straight forward manner is that the number of rules mined becomes extremely large. The rule mining will with a relatively low support threshold most probably produce rules covering almost all possible combinations of events. Hence, the key contribution of this thesis is to do as many reasonable choices possible, in order to restrict the rule mining process in a way that the smallest possible rule set interesting for prediction is obtained.

As already introduced, this thesis has outlined a two step process consisting of rule mining and rule selection. The focus of this thesis is at enforcing constraints in order to restrict the rule discovery process. The rule syntax and semantics defined in Chapter 3 defines a set of constraints which should be used to target rules suited for prediction. The restricted rule mining approach in Chapter 4.3 also introduces a $lift > 1$ constraint. The rule selection process derived in Chapter 5 also introduces three new constraints. For the sake of explicitness these constraints are summed up according to where they were introduced:

Constraints introduced in Chapter 3:

1. Single event in the consequent
2. Mutiple events in the antecedent
3. Serial ordering between antecedent and consequent
4. Parallel antecedent
5. Maximum time of validity
6. Minimum reaction time of operators

Constraints introduced in Chapter 4:

7. Minimum support
8. Lift > 1

Constraints introduced in Chapter 5:

9. Rules which do not influence a given target event are discarded
10. Minimum improvement required for specification rules
11. Rules are ranked based on J-measure, the top ranked rules are selected and submitted to an expert

With this set of constraints in mind, two approaches are possible: 1) Using standard rule mining algorithms and enforcing constraints in a post-processing phase. 2) Using a restricted rule mining algorithm to efficiently enforce some of the constraints, and enforce the rest during post-processing. These two different approaches are outlined respectively as a simple and an advanced solution to the problem of mining rules suited for prediction from event sequences.

6.2.1 Simple solution in ConocoPhillips

Motivated by the discussion in Chapter 3.4, a simple solution using standard procedures in SAS Enterprise Miner 5.2 is of great interest to ConocoPhillips. As stated in Chapter 3.4, SAS Enterprise Miner 5.2 supports association rule mining and sequence rule mining. Summarizing the suggested methods for implementation in ConocoPhillips from Chapters 3.4, 4.4 and 5.4 the solution outlined in Algorithm 3 is suggested for easy implementation in ConocoPhillips. Note that the time information included in the rule syntax, and enforcement of the minimum reaction time constraint, are left out.

Algorithm 3: Simple solution in ConocoPhillips

```
1.  Convert event sequence to transactions by using the technique of Chapter 4.1.1 ;
2.  * Phase 1 : Rule mining * ;
3.  Apply the association rule mining procedure in SAS Enterprise Miner 5.2 ;
4.  * Phase 2 : Rule selection * ;
5.  Remove rules with $lift \leq 1$ ;
6.  Remove rules with more than one event in the consequent ;
7.  Do a pass over the data to ensure the correct temporal ordering of rules ;
8.  Remove rules which do not influence the target event by using the method outlined in
    Chapter 5.4 ;
9.  Remove specification rules which do not satisfy the $min\_imp$ constraint by using the
    method outlined in Chapter 5.4 ;
10. Compute J-measure for all rules based on the formula described in Chapter 5.4 ;
11. Sort rules based on J-measure ;
12. Select the top $X$ rules for expert validation ;
```

As already stated, such an approach is not the theoretically optimal solution because constraints, e.g. temporal ordering of the rule, may be more efficiently enforced during rule mining. Enforcing these constraints in a post-processing phase can be equally or more resource demanding than the rule mining process itself. It will result in a significantly larger amount of rules to post-process. This implies that complexity is shifted from the rule mining algorithm to the post-processing phase. Operations similar to operations done in the rule mining algorithm must be done in the post-processing phase, introducing redundancy and an increase in resource usage and running times. However, in the ConocoPhillips case these operations are only done once, and the running time is probably irrelevant, as long as it is within weeks. Because SAS Enterprise Miner 5.2 is the preferred implementation environment in ConocoPhillips, such an approach is acceptable, but in general such an approach is unnecessary complicated and inefficient.

6.2.2 Advanced solution

Based on the rule syntax and semantics defined in Chapter 3 a restricted rule mining algorithm, where the amount of redundancy in computations is minimized, was suggested in Chapter 4.3. Those constraints that efficiently can be enforced during rule mining are enforced in the restricted rule mining algorithm. Those constraints more naturally enforced separately are part of the rule selection phase. Summarizing the suggested methods from Chapters 4.3 and 5.3, the solution outlined in Algorithm 4 is a complete solution for efficient discovery of restricted association rules suited for prediction from event sequences. Note that time information for rules, and the minimum reaction time constraint, are enforced in the restricted rule mining algorithm.

Algorithm 4: Advanced general solution

```
1. * Phase 1 :  Rule mining * ;
2. Apply the restricted rule mining procedure described in Chapter 4.3 ;
3. * Phase 2 :  Rule selection * ;
4. Remove rules which do not influence the target event by using the method outlined in
   Chapter 5.3 ;
5. Remove specification rules which do not satisfy the $min\_imp$ constraint by using the
   method outlined in Chapter 5.3 ;
6. Sort rules based on J-measure ;
7. Select the top $X$ rules for expert validation ;
```

6.3 Validity of method

The foundation of this thesis is the well proven association rule mining paradigm. However, by extending this paradigm to sequences of events, and by incorporating various constraints, the premises of the method are altered.

6.3.1 Enforcing constraints

In both the restricted rule mining process and the rule selection process, constraints on the rules are enforced in order to target the resulting rule set towards prediction. This implies that some rules that normally would have been a part of the rule set, if standard association rule mining methods are used, will be removed or left out. For reasoning and prediction this poses an interesting issue. For an example consider Figure 6.1.

If a reasoning network is built from the rule set it is possible that two dense sub-graphs have been constructed, only linked together by one single rule(Y). If this rule is removed due to some constraint, the two sub-graphs are split into two unconnected graphs, and the sub-graph not containing the target event is discarded later in the process. Hence, a large number of rules that was connected with the target event are removed due to the removal of one rule, based on a defined constraint. Another issue is that by removing rules from a complete rule set more and more information is lost. This has implications for reasoning and prediction because the soundness of e.g. reasoning can be broken by removing certain rules. Hence, the implications of each constraint should be discussed with respect to validity of reasoning

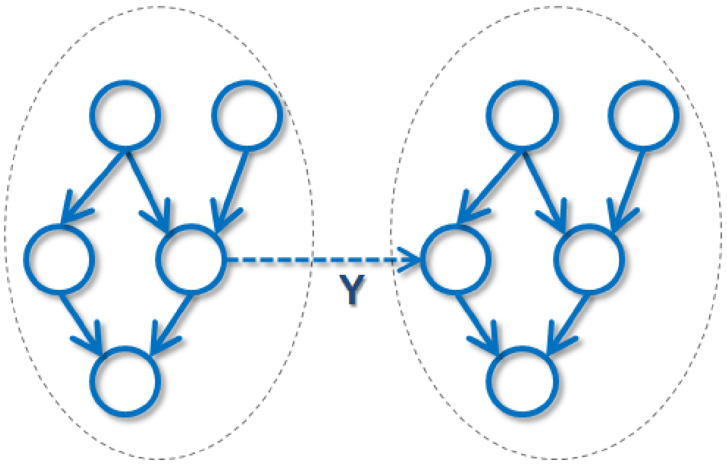


Figure 6.1: A single rule combining two dense sub-graphs

and prediction.

Those constraints developed from the rule syntax and semantics have been evaluated with respect to reasoning and prediction. Only rules with a structure not relevant for prediction are removed. Based on the work done by Helland [20], reasoning structures for prediction usually comes in the form of a network ending in a single target node. The rule syntax and semantics actually ensures that only rules that are candidates for such a network are targeted. Hence, information relevant for prediction is not lost. The same argument applies to the rule selection task that excludes rules which cannot be combined in a way that they influence the target event. However, the additional constraints introduced in the rule mining and rule selection process may influence the soundness of reasoning. Using the list of constraints from last sub-chapter, these constraints are 7, 8, 10 and 11.

Constraint 7: The minimum support constraint, which is an integrated part of all A-priori based algorithms, must be considered with care. One particular problem is related to the low support of failure events. It should be clear that rules concerning the targeted failure event are relevant with respect to prediction. These rules are bounded downwards by the support of the failure event, which often reside at the boundary of minimum support. Because of this, rules that are highly interesting for prediction may easily fall below the minimum support threshold. If a large number of the rules regarding the failure event are discarded by the minimum support constraint, the validity of the prediction may be questioned. E.g. How is it possible to state that given only a small subset of all rules concerning the target, the probability of the failure event is 60%? The validity of such a statement increases with the fraction of possible rules concerning the target that is included in the reasoning. Hence, the selected threshold of support should be set at a significant lower level than the support of the targeted failure event.

Constraint 8: In Chapter 4, it was suggested to enforce a constraint on the $lift$ of rules suited for prediction. As discussed, this constraint is in general valid for prediction because negative correlations are not interesting. However, analogue to the case of minimum improvement depicted in Chapter 4.2.2, specification rules with negative improvement, and perhaps negative $lift$, provide information to the reasoning process. If an event that greatly increases our belief in the target event is observed we will have a high belief in the target event. If then an observation of another event is done that lowers our belief in the target, then the specification rule should take over. This specification rule should in theory be included even if it has a $lift$ lower than 1. Even if it lowers our belief in the target below the probability of the target occurring alone, it contradicts the single observation of the high lift rule. In practice this is a unlikely scenario, because if the confidence of a single rule is high it is nearly impossible to have a specification rule with a confidence that approaches the low support which single events have.

Constraint 10: Enforcing the minimum improvement constraint of Chapter 5 is a more complex discussion. By removing specification rules it is intuitive to believe that global information is lost. However, the only impact a specification rule will have on the reasoning process is if it provides new information about the consequent. Consider the case where the rules $A \rightarrow C$ and $B \rightarrow C$ with the same confidence exists. Then if either A or B, or both, are observed we will have a given belief in C. If then the rule $A, B \rightarrow C$ is introduced, with the same confidence as the two other rules, our belief in C remains unchanged. Belief in C is measured by $lift$, but with a fixed consequent lift and confidence only differs by a constant. Hence, it is valid with respect to prediction to discard the specification rule from the rule set based on confidence.

Constraint 11: All constraints enforced until now have been shown to not, or only marginally, influence the validity of reasoning and prediction based on the rule set. This corresponds to the goal of obtaining a rule set well suited for prediction, because a large amount of redundant information is removed. When it comes to the last aspect of the rule selection process, selecting a subset of rules based on a ranking done by J-measure, the reasoning process is heavily influenced. The choice of interestingness measure determines the premises of the following reasoning structure. If a subset of all possible rules is selected by some interestingness measure, and used as the rule base, then the following reasoning structure must be evaluated for soundness and completeness with respect to the chosen interestingness measure. The conclusions reached by reasoning on a subset of all rules may need to be moderated, or not necessarily regarded as absolute truths. This discussion is elaborated in the work by Helland [20].

By removing a large number of rules by using an interestingness measure, the resulting rule set also loses the property of being complete. This can break the validity of the reasoning process because not all theoretically available information is used as basis for reasoning. However, as already stated, indicates preliminary tests on example data from ConocoPhillips that approximately 143 000 rules will still be present in the rule set after enforcing all constraints except minimum improvement and J-measure selection. See Appendix A for details. Considering this amount of rules, it is clear that a more drastic approach is required to get a rule set that is manageable by an expert. Hence, the discussion of global validity becomes subordinate, and a more pragmatic view is taken. This also correspond better with the goal of obtaining a reasoning structure that can be comprised of the most prominent and interesting rules, which can be showed to the operators as a graphical expert system.

It should be stressed that the techniques proposed in this thesis are not intended to be fully automatic. The validation done by an expert is a crucial step, but can be time consuming. Assisting the expert in selecting the interesting rules from a large rule set is not the focus of this thesis, but various techniques have been developed in the literature for interactive rule selection. One example is the rule exploration tool in TASA developed by Manilla et al. [25]. Here experts are given the possibility to explore a huge rule base by specifying patterns or thresholds on different measures related to the rules. This allows the expert to quickly prune a large set of rules and quickly target interesting rule types.

6.3.2 The problem of low support

After analysing example data received from Torulf Mollestad concerning the 2/4 J process described in Chapter 1, a crucial observation was made. The failure event targeted for prediction occurred 22 times during one year of data. When compared with other event types in the data, occurring frequently and within short time intervals, the failure event gets a much smaller support than all other events. This implies that using A-priori based rule mining algorithms, the minimum support threshold needs to be set far less than the support of the failure event, if any patterns involving the target event should be found. This results in a support threshold that hardly confines the search space, which again results in a exponential explosion of number of candidate episodes for each level of the algorithm. This problem, and the fact that the event data is rather dense, results in an extremely large number of rules.

However, a large part of this problem comes from the low support threshold of the target event with respect to the whole data set. No solution to this problem has been outlined in this thesis, but one possible technique is to select a sub-set of the data within a defined time interval before each target event. If for example, a time interval of one day before each target event is selected, and this constitute the event sequence for rule mining, a larger support threshold can be set, and the algorithm will operate on less data. This will also result in a more confined search, and discovery of less patterns in the data. Such an approach can probably capture the interesting patterns with respect to failure, but the distribution of events is wrong with respect to the data. Hence, the actual support, confidence and lift of rules must be computed by looking at the original dataset. Other possible solutions include selecting both intervals before the target event and random intervals, sampling with a bias towards the time around failure or other similar techniques. However, such approaches are highly questionable, and have not been considered in this thesis because mining on the full dataset proved to be feasible.

6.3.3 Issues of uncertainty

Some of the choices and assumptions made in this thesis have been taken from a theoretic point of view with a degree of uncertainty.

The uncertain issues in this work are:

- A theoretical based decision of using rules with a parallel antecedent
- A theoretical based choice of using J-measure to rank rules

If a complete reasoning structure had been available, an empirical conclusion of which type of parallel or serial antecedent to choose could have been reached by measuring the predictive performance of the two alternatives on actual process data. The same approach could have been used to test the various interestingness measures outlined in Chapter 5.1, and accepting or rejecting the hypothesis that J-measure performs better than other available interestingness measures for an initial selection of rules. Another empirical approach for selecting the correct interestingness measure could have been done if an domain expert were available. The expert could rank a set of rules, and then a ranking of the same rules can be done by using all of the interestingness measures described in Chapter 5.1 and compare their suggested ranking with the experts ranking. The measure which produces the best match can be selected as the preferred interestingness measure.

6.4 The overall approach

As introduced in Chapter 1, this thesis is a part of a larger three step process for data utilization in ConocoPhillips defined by Torulf Mollestad. The underlying idea of this process is that events can be defined from sensor data, and that an association rule mining approach can be applied to the resulting event sequence to discover association rules. The idea is that these rules can be combined in a way which allows for pre-warning of oil production failures.

This three step process is novel, and to the authors knowledge no such approach to pre-warning in process systems exists in the literature. There exist other approaches in the literature for warning about failures in process systems based on events, but with different focus. These focus on building a model characterizing normal behaviour of the system, and detecting conflicts with this behaviour. One example is Nielsen et al. [36] which learn a Bayesian network from sensor data that describe the normal behaviour of the system. Another more similar example is Yairi et al. [45] which define events from sensor data using a clustering approach and mine association rules based on these events. These association rules are used as a model of normal behaviour, and conflicts with this behaviour is detected real-time.

Another related work is the TASA system developed by Hatonen et al. [19]. Here association rules are mined from alarm events in a telecommunication system. The focus is not on prediction, but removing redundant alarms and generalizing low level alarms. This approach has shown very good results, and proves that association rules can be efficiently discovered from event sequences, and be applied in real-time systems.

The strength of using an association rule mining approach for prediction in the oil production domain is mainly its intuitive appeal, and the possibility of combining large amounts of data and an automated process with expert knowledge. Association rules provide an intuitive expression of relationships between events, and allows for individual interpretation and validation of a rule by an expert. By combining association rules in a visual way, explanations of warnings can be given to operators in a way that is easy to understand. Another reason for using the association rule mining paradigm for prediction of oil production failures is the great amount of data gathered from sensors. Other model learning techniques, such as Bayesian network learning, cannot handle the amount of data in question. By transforming sensor data to time based events, the amount of data is reduced, and the association rule mining paradigm provides an efficient method for discovering dependencies between events.

Reasoning based on association rules does not correspond to a mathematically correct model for reasoning, such as learning Bayesian networks. As noted in [28], association rules essentially describe a local property of the data (the dependency between two events), but they also provide global information about the global dependencies in the data. This discussion is further elaborated in the work by Helland [20]. The conclusion is that reasoning based on association rules should give an reasonable approximation of a globally valid method, which can increase the probability of pre-warning future oil production problems in ConocoPhillips.

As concluded in the recent project thesis by Helland and Christiansen [11] there exists no answer in the literature to whether the three step process is feasible or not, but by using the theoretical framework derived in this thesis the rule mining and rule selection process is feasible. The work by Helland [20] concludes that combining association rules in a reasoning structure can give valuable information for pre-warning oil production failures. The open questions then becomes whether the user defined events defined in the first step of the process are the correct events to best describe oil production failures. The rule mining phase will not perform better than the quality of the input, and the resulting predictive performance will be equally influenced. Hence, it becomes important to derive events in close cooperation with domain experts to ensure the validity and quality of events.

6.5 Rule mining in ConocoPhillips

Going back to the problem case description of the 2/4 J process in Chapter 1, where the work done by Torulf Mollestad defined a three step data mining vision, the first step can be considered as solved by the work done by Torulf Mollestad in ConocoPhillips. The result of Torulf Mollestad's work is events with a clear formalism describing important properties of sensor data. Events which do not correlate with the specified target event are removed, ensuring a smaller and more relevant dataset.

The suggested framework of this thesis can be implemented directly in ConocoPhillips in the way specified in Chapter 6.2.1. It will then operate directly on the resulting event sequence from Torulf Mollestad's work. This allows for the use of SAS Enterprise Miner 5.2 to do all tasks. As an alternative the advanced solution can be implemented outside Enterprise Miner, e.g. in Java, and the resulting rules can be imported in Enterprise Miner. The rule selection process can then be done in SAS Enterprise Miner 5.2.

After ranking the rules by J-measure, a relatively small set of rules can be validated by process experts, and perhaps offshore operators with the desired amount of experience. The focus should be on selecting those rules which are believed to hold in the real world, and that are believed to possibly predict failures. The result will be a small set of rules that can be combined in a way that allows for real-time reasoning and prediction, which is shown to be feasible in the parallel work by Helland [20].

The contribution of this work to the ongoing research in ConocoPhillips is a clear specification of which properties rules suited for prediction should hold, in the form of a defined rule syntax and rule semantics. In addition, this work shows how it is possible to do a full scale rule mining process on the time series data available from the 2/4 J process, and how this process can be integrated with a rule selection approach to obtain a small set of rules suited for prediction. This work supports the three step process for going from sensor data to an operational system developed in ConocoPhillips by Torulf Mollestad.

Chapter 7

Conclusion and further work

In this thesis methods have been derived for discovering event patterns in time series, called restricted association rules, in order to pre-warn about future problems in oil production processes. It corresponds to the second step of discovering dependencies between events in a guideline for data utilization in the oil production domain developed by Mollestad [32, 33]. Three sub-problems of the rule mining step have been explored:

- Defining characteristics for rules suited for prediction
- Restricted association rule mining
- Rule selection

Standard association rule mining is developed for market basket analysis where data appear as transactions. Generalizing this approach to handle event sequences, and infrequent failure events, result in a vast number of rules. With the goal of building an expert system which reason based on the discovered rules it becomes important to reduce the number of rules to a size manageable for human validation and exploration. By considering the specific task of predicting time based events a more restricted rule discovery process can be derived. Rules suited for prediction of time based events differ from regular association rules, and by identifying key characteristics of such rules a restricted rule syntax and rule semantic can be defined. Some important properties of rules suited for prediction are: single consequent, multiple antecedent, serial ordering between antecedent and consequent, parallel ordering in antecedent and a defined maximum time of validity.

Based on a restricted rule syntax a two step process for obtaining a small set of rules suited for prediction can be derived. This process consists of a restricted rule mining phase, and a rule selection phase where redundant rules are removed. Rules satisfying the defined rule syntax can be efficiently discovered by using a customized algorithm derived in this thesis, based on the concept of minimal occurrences. As an alternative for easy implementation in ConocoPhillips, standard association rule mining algorithms can be used, and the rule syntax can be enforced in a post-processing phase.

Besides enforcing the restricted rule syntax, redundant rules can be removed by removing rules with a $lift \leq 1$, removing specification rules which brings no new information and by removing rules which when combined do not influence the target event. The rule set obtained after these selections will in the ConocoPhillips case still be to numerous to handle. As a solution to this an evaluation of different interestingness measures was done, and the J-measure is chosen as the preferred measure for ranking rules. J-measure is founded on the concept of information theory which has been proven to give good predictive performance in other domains. A subset of the top ranked rules can be submitted to an expert together with their support, confidence and lift values for validation and selection.

Due to its simplicity and intuitive approach the association rule mining paradigm is considered as the preferred choice of learning from data. Another important issue is the efficiency needed for handling large amounts of sensor data. Association rule mining provides a computational efficient solution to discover dependencies in large amounts of data. The parallel work by Helland [20] shows how the set of restricted association rules can be combined in a reasoning structure, which is believed to give good predictive performance.

This thesis has shown that it is theoretically feasible to discover restricted association rules from sensor data. It is concluded that one of the two suggested solutions can be implemented in ConocoPhillips with a large probability of being able to pre-warn about future production anomalies in the 2/4 J process. For further work, a natural continuation is testing both serial antecedent and parallel antecedent on the actual process data, and estimating their predictive performance. Also, the different interestingness measures described in Chapter 5.1 should be tested on actual data, and their predictive performance estimated. This can give a definitive answer to which interestingness measure that performs best for the task at hand. Further, an exploration of the possibility of integrating the rule selection process with the network learning phase of Helland [20] can be done. These two tasks overlap, and redundant rules can possibly be more efficiently removed during network learning.

Bibliography

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993. 2.1, 2.1, 1, 5.1.2

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994. 2.1

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In P. S. Yu and A. S. P. Chen, editors, *Eleventh International Conference on Data Engineering*, pages 3–14, Taipei, Taiwan, 1995. IEEE Computer Society Press. 3.3.1, 4.1

[4] A. Agresti. *Categorical data analysis*. Wiley, New York, 1990. 5.1.2

[5] ConocoPhillips - 2006 Annual Report. http://www.conocophillips.com, 30 May 2007. 1.2

[6] J. Blanchard, F. Guillet, R. Gras, and H. Briand. Using information-theoretic measures to assess association rule interestingness. In *5th IEEE International Conference on Data Mining (ICDM'05)*, pages 66–73. IEEE Computer Society Press, 2005. 5.1

[7] J. Blanchard, F. Guillet, R. Gras, and H. Briand. Using information-theoretic measures to assess association rule interestingness. In *Proceedings of the fifth IEEE International Conference on Data Mining ICDM'05*, pages 66–73. IEEE Computer Society, 2005. 5.1.3

[8] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984. 5.1.2

[9] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *SIGMOD '97: Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, pages 255–264. ACM Press, 1997. 2.1, 3.2.2, 5.1.2, 5.1.2

[10] S. Chawla, B. Arunasalam, and J. Davis. Mining open source software (OSS) data using association rules network. Technical Report TR535, School of IT, University of Sidney, Sidney, NSW, Australia, 2003. 5.2

[11] J. Christiansen and P. K. Helland. Mining time series in order to predict loss in oil production. Technical report, Norwegian University of Science and Technology, 2006. 1.3.3, 3.2.1, 4, 6.4

[12] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991. 5.1.3, 5.1.3

[13] G. Das, K. Lin, H. Mannila, G. Renganathan, and P. Smyth. Rule discovery from time series. In *In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*. AAAI Press, 1998. 4.1

[14] J. de Jongh and ConocoPhillips. Business and Company Background COP, internal document. 2006. 1, 1.2, 1.3

[15] F. H. Fossan. "Masteroppgave - domenespørsmål". Personal email. Fredrik Høymer Fossan is a Maintenance Optimisation Engineer at ConocoPhillips. The mail is dated 11 May 2007. 3.3.1

[16] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. *ACM Comput. Surv.*, 38(3):9, 2006. 5.1

[17] M. Hahsler and K. Hornik. New probabilistic interest measures for association rules. Report 38, Research Report Series, Department of Statistics and Mathematics, Wirtschaftsuniversität Wien, August 2006. 5.1.2, 5.1.2

[18] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.-C. Hsu. Freespan: frequent pattern-projected sequential pattern mining. In *KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 355–359. ACM Press, 2000. 4.1

[19] K. Hatonen, M. Klemettinen, H. Mannila, P. Ronkainen, and H. Toivonen. Knowledge discovery from telecommunication network alarm databases. In S. Y. W. Su, editor, *Proceedings of the twelfth International Conference on Data Engineering, February 26–March 1, 1996, New Orleans, Louisiana*, pages 115–122. IEEE Computer Society Press, 1996. 6.4

[20] P. K. Helland. A survey of combining association rules for pre-warning of oil production problems. Master's thesis, Norwegian University of Science and Technology, 2007. 1.3.3, 3.2.3, 3.3.2, 5.2, 6.1, 6.3.1, 6.4, 6.5, 7, A.2

[21] M. L. Hetland. *Evolving sequence rules*. PhD thesis, Norwegian University of Computer and Information Science, 2003. 5.1.3

[22] R. Hilderman and H. Hamilton. Knowledge discovery and interestingness measures: A survey, 1999. 5.1

[23] J. Hipp, U. Güntzer, and G. Nakhaeizadeh. Algorithms for association rule mining — a general survey and comparison. *SIGKDD Explorations*, 2(1):58–64, 2000. 2.1, 2.1

[24] S. Institute Inc. World Headquarters. SAS Enterprise Miner 5.2 - Fact sheet. Last visited 4 May 2007. 3.4

[25] M. Klemettinen, H. Mannila, and H. Toivonen. Interactive exploration of interesting findings in the telecommunication network alarm sequence analyzer (tasa). *Information & Software Technology*, 41(9):557–567, 1999. 6.3.1

[26] S. Kullback. *Information theory and statistics*. John Wiley and Sons., New York, 1959. 2.2

[27] B. Liu, W. Hsu, S. Chen, and Y. Ma. Analyzing the subjective interestingness of association rules. *IEEE Intelligent Systems*, 15(5):47–55, 2000. 5.1

[28] H. Mannila. Local and global methods in data mining: Basic techniques and open problems. In *ICALP '02: Proceedings of the 29th International Colloquium on Automata, Languages and Programming*, pages 57–68, London, UK, 2002. Springer-Verlag. 6.4

[29] H. Mannila and H. Toivonen. Discovering generalized episodes using minimal occurrences. In *Knowledge Discovery and Data Mining*, pages 146–151, 1996. 3, 3.1.2, 3.2.3, 4.1, 4.1.3

[30] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovering Frequent Episodes in Sequences. In U. M. Fayyad and R. Uthurusamy, editors, *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, Montreal, Canada, 1995. AAAI Press. 3, 3.1.2, 3.1.2, 3.2.3, 4.1, 4.1.2

[31] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1(3):259–289, 1997. 3, 3.1.2, 4.1.2, 4.1.3, 1, 4.1.3

[32] T. Mollestad. ConocoPhillips (COP) - Event modelling. Forthcoming. 1.3.3, 1.3.3, 7

[33] T. Mollestad. Warning process problems using an event-based interpretation to time series. Forthcoming. 1.3.3, 1.3.3, 7

[34] M. N. Moreno, S. Segrera, and V. F. López. Association rules: Problems, solutions and new applications. In *III Taller Nacional de Minería de Datos y Aprendizaje, TAMIDA2005*, pages 317–323, 2005. 2.1

[35] F. Mosteller. Association and estimation in contingency tables. *Journal of the American Statistical Association*, 63(321):1–28, 1968. 5.1.2, 5.1.2

[36] T. D. Nielsen and F. V. Jensen. Alert systems for production plants: A methodology based on conflict analysis. In *ECSQARU*, pages 76–87, 2005. 6.4

[37] G. Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, pages 229–248. AAAI/MIT Press, 1991. 2.1, 5.1.1, 5.1.2, 5.1.2

[38] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986. 5.3

[39] J. Roberto J. Bayardo, R. Agrawal, and D. Gunopulos. Constraint-based rule mining in large, dense databases. *Data Min. Knowl. Discov.*, 4(2-3):217–240, 2000. 4.2, 4.2.2

[40] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–, july, october 1948. 2.2, 2.2, 2.2, 2.2, 2.2, 5.1.3, 5.1.3

[41] P. Smyth and R. M. Goodman. An information theoretic approach to rule induction from databases. *IEEE Transactions on Knowledge and Data Engineering*, 4(4):301–316, 1992. 5.1.3, 5.1.3

[42] R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In D. Heckerman, H. Mannila, D. Pregibon, and R. Uthurusamy, editors, *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining, KDD*, pages 67–73. AAAI Press, 14–17 1997. 4.2.1

[43] O. T. Stephane Lallich and E. Prudhomme. Association rule interestingness: measure and statistical validation. In J. H. Hamilton and F. Guillet, editors, *Quality Measures in Data Mining*, pages 251–276. 2007. 3.2.2, 5.1, 5.1.1, 5.1.2

[44] P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the right objective measure for association analysis. *Information Systems*, 29(4):293–313, 2004. 5.1, 5.1.1, 5.1.2, 5.1.3

[45] T. Yairi, Y. Kato, and K. Hori. Fault detection by mining association rules from house-keeping data. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space, 2001.*, 2001. 6.4

[46] M. J. Zaki. Sequence mining in categorical domains: incorporating constraints. In *CIKM '00: Proceedings of the ninth international conference on Information and knowledge management*, pages 422–429. ACM Press, 2000. 4.1

Appendix A

Prototype testing

In order to validate the restricted rule mining approach in Chapter 4.3, a prototype for restricted rule mining has been implemented. This prototype is highly experimental, but serves as a proof of concept for the ideas presented in this thesis. Implementation is not a part of this assignment, and because of this the description of the prototype and the obtained results are presented in a short and informal way. This appendix will describe the implementation specific details of the prototype, and show some results obtained by testing on actual data from the 2/4 J case. This should serve as an indicator on how the suggested solutions will perform on actual data, and what to expect of the results.

A.1 The prototype

The prototype is implemented in Java 6.0 by transforming Algorithm 2 from Chapter 4.3 to Java code in a straight forward way. Classes are implemented for `Episodes`, `Rules` and `Intervals`. These objects are just holding objects, where `Intervals` only consists of two integers representing the start and end time of an occurrence. `Episodes` has an array list of `Intervals` (occurrences) and an array of identifiers denoting which events the episode consists of. In addition an `EpisodeList` is introduced to hold episodes discovered in each iteration of the rule mining algorithm. This list has a list structure and a hashmap mapping events and episode occurrences for easy lookup when sub-episode checking is done.

Checking whether an episode is frequent can be done easily by evaluating the size of the `Intervals` array. Whether a sub-episode of a given episode is frequent can be checked by looking up the sub-episode in the $(k-1)$ `EpisodeList`. If it exists in this list it is frequent. Rest of the algorithm is implemented as described in Algorithm 2.

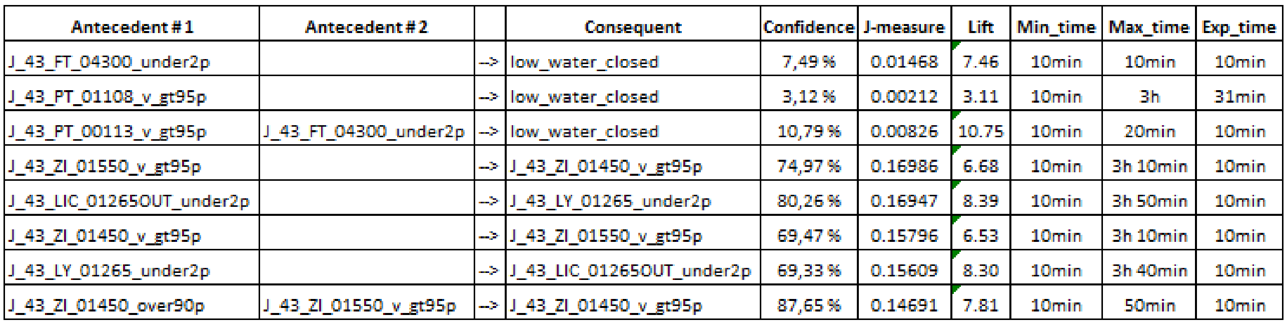
This prototype holds all episodes of the current and preceding iterations in main memory. At each iteration, rules of size k are generated from the k'th level of frequent episodes. This makes it possible to discard all episodes which are not a part of the current or preceding iteration, because they will not be needed. This results in a reduced memory usage, but still all episodes and occurrences for the current and the preceding iterations need to reside in main memory. For large datasets this results in an increased use of memory. The prototype is not attached with this thesis, but can be obtained from the author if needed.

A.2 Test results

Test data is obtained from the 2/4 J process described in Chapter 1.3.2. This data has been selected and preprocessed by Mollestad according to the method described in Chapter 1.3.3. The resulting dataset from Mollestad is 5,6 Mega Bytes with data, residing in a flat text file, on the form <*event,time*>. Where *event* is a textual description of the occurring event, and the *time* is the time of occurrence of the event. An example is: *J_43_TT_20854CU_under2p,1449014400*. The dataset consists of 168 162 occurrences during one year of data. The event targeted for prediction is the event *"low_water_closed"*. This event occurs 22 times in the data set.

The implemented prototype is executed on the dataset with the following parameters: *minimum reaction time = 10 minutes*, *maximum time of validity for a rule = 4 hours*, *frequency threshold = 10 occurrences*, and restricted to 3 iterations (2 events in the antecedent). The running time is approximately 10 minutes and the memory usage peaks at approximately 1,5 Giga Bytes on a 3Ghz Pentium 4 computer with 3 Giga Bytes of RAM. It should be noted that the memory usage is artificially high because this is an experimental prototype not developed for efficiency and scalability, but as a proof of concept.

The resulting rule set consists of 443 363 rules sorted by the J-measure. A screen dump of some example rules from the rule set is given in Figure A.1.



Antecedent # 1	Antecedent # 2		Consequent	Confidence	J-measure	Lift	Min_time	Max_time	Exp_time
J_43_FT_04300_under2p		-->	low_water_closed	7,49 %	0.01468	7.46	10min	10min	10min
J_43_PT_01108_v_gt95p		-->	low_water_closed	3,12 %	0.00212	3.11	10min	3h	31min
J_43_PT_00113_v_gt95p	J_43_FT_04300_under2p	-->	low_water_closed	10,79 %	0.00826	10.75	10min	20min	10min
J_43_ZI_01550_v_gt95p		-->	J_43_ZI_01450_v_gt95p	74,97 %	0.16986	6.68	10min	3h 10min	10min
J_43_LIC_01265OUT_under2p		-->	J_43_LY_01265_under2p	80,26 %	0.16947	8.39	10min	3h 50min	10min
J_43_ZI_01450_v_gt95p		-->	J_43_ZI_01550_v_gt95p	69,47 %	0.15796	6.53	10min	3h 10min	10min
J_43_LY_01265_under2p		-->	J_43_LIC_01265OUT_under2p	69,33 %	0.15609	8.30	10min	3h 40min	10min
J_43_ZI_01450_over90p	J_43_ZI_01550_v_gt95p	-->	J_43_ZI_01450_v_gt95p	87,65 %	0.14691	7.81	10min	50min	10min

Figure A.1: Example of rules from the resulting rule set

The number of rules discovered are approximately 4 times larger than the number of occurrences of events. This is because of the low support threshold needed to capture rules concerning the target event. With a support threshold on 10 occurrences, a large number of combinations of events is discovered. This corresponds to the discussion of Chapter 6.3.2. Even with the number of rules discovered, this testing on real process data indicates that mining restricted rules from event sequences is feasible using the method derived in Chapter 4.3.

Similar testing of the rule selection phase of the outlined solution has not been performed, but the step of removing rules that do not influence the target event has been performed using a simple mock up algorithm based on the work of Helland [20], and the suggested solution outlined in Chapter 5.2. After performing this step the rule set consists of approximately 143 000 rules.