Chapter 1
Introduction

The idea of extracting knowledge from sets of data emerged back in the 90s, motivated by the decision support problem faced by several retail organizations that due to several technological advances, were able to store massive amounts of sales data. At that point, the research field of knowledge discovery started as an active area of investigation, and data mining, one of the most challenging steps of the process, was meant to provide efficient algorithms and techniques to automatize the exploratory analysis of the data. In its simplest form, such data is viewed as a set of transactions where each transaction is a set of items (attributes of the database), that is, a simple binary relation. This representation is known popularly as market-basket data.

One natural way of representing knowledge in this context is to look for causal relationships, where the presence of some facts suggests that other facts follow from them. One of the reasons for the success of the association rule framework is that in the presence of a community that tends to buy, say, sodas together with the less expensive spirits, a number of natural ideas to try to influence the behaviour of the buyers and profit from the patterns, easily come up. Approaches to find such associations started long before the area of knowledge discovery became so popular. For example, Duquenne and Guigues in [46] and also Luxenburger in [93], studied bases of minimal nonredundant sets of rules from which all other rules can be derived. The former studied these bases for association rules with 100% confidence, and the latter association rules with less than 100% confidence, but neither of them considered the support of the rules, i.e. the number of transactions in the data supporting the rule. Nowadays, this task is widely known as the association rule mining problem, and became very popular since it was reformulated by Agrawal et al. in [3]. The reformulation made by Agrawal et al. introduced this notion of support, allowing for the pruning of those rules whose number of occurrences in the data was not over a user-specified threshold.

Taking this association rule mining problem, there is a rich variety of algorithmic proposals whose strategy is to look for the frequent itemsets in the data, i.e., those sets of items with numbers of occurrences over a threshold, and then, constructing implications between these discovered frequent itemsets. The most well-known of these algorithms is Apriori [4]; it traverses the search space in a breadth-first fashion,

G.C. Garriga: *Formal Methods for Mining Structured Objects*, SCI 475, pp. 1–11.
DOI: 10.1007/978-3-642-36681-9_1

using the antimonotonicity property of the support to prune unnecessary candidates. After this first proposal, many other algorithmic strategies and methods emerged to improve the efficiency of Apriori: e.g. some of them suggested new structures to compact the original database into main memory, others proposed a way to traverse the search space in a best-first fashion, or performing the mining over a sample of the original transactions instead of all the data, or even some publications worked with parallel algorithms. Among many others, the following works are relevant [2, 20, 9, 10, 27, 28, 64, 69, 74, 72, 77, 91, 90, 101, 112, 123, 142, 145, 138, 147]. For a recent survey on the algorithmic trends of this problem see [71]. To complement these algorithmic advances with theory, in [66, 67] the connection between association rules and hypergraph transversals was presented; the authors also gave complexity bounds to discover those maximal sets with their hypergraph formulation.

Soon after the publication of the aforementioned algorithms, the following problem was how to reduce the huge number of association rules that were extracted by the algorithms. Different criteria were needed to make a judgment whether the extracted implication contained useful information. A classical way to rank the final rules is by means of statistical metrics (e.g. confidence [3, 93], conviction [27], lift [26] and so on). There are a large number of proposals as to how to measure the strength of implication of a rule, yet criticisms of various forms can be put forward for any measures; e.g. one of the criticisms for lift is its symmetry, which makes it impossible to orient rules. Surveys, with appropriate references, are given in [21, 22, 55, 48, 76, 115, 119, 127]. Recent advances on defining the significance of itemsets and rules are, among many others, e.g. [49, 121, 94, 130].

A complementary approach to ranking rules with statistical metrics consists of generating a basis of association rules from which the rest can be derived. As mentioned before, this approach was initially studied by Duquenne and Guigues in [46], and later by Luxenburger in [93]; yet, they did not consider any notion of minimum support for the rules. This idea evolved towards considering only those frequent *closed itemsets* instead of all the frequent itemsets when first mining the data, and after that, generate only those rules indicated by the closure system (see e.g. [18, 39, 102, 120, 126, 133, 144, 137, 141]). Using a similar idea, the work in [40] introduced a new rule of inference and defined the notion of association rule cover as a minimal set of rules that are non-redundant with respect to this new rule of inference. Other complementary ideas are the non-derivable itemsets of [31, 32], relying on a complete set of deduction rules. More recent works are [12, 11, 82]. Even if these approaches of covering rules or compressing patterns also result effective in practice, we will focus here on the theory related to closed patterns.

As we shall see, closed itemsets are particularly interesting from a theoretical point of view due to their mathematical foundations based on formal concept analysis and concept lattices. Broadly speaking, this theory is based on the definition of a Galois connection for a binary relation between a set of objects and a set of items, that is, the original data. This Galois connection enables a closure system, i.e. a complete lattice (a Hasse diagram) of formal concepts. Each one of these concepts captures the information of closed itemsets, hence implications, in the data.

The theory of Galois lattices has proved to be an expressive technique to reason about the binary data.

Nonetheless, for many real applications data are represented in more complex structures, such as sequences, trees or graphs. Squeezing these structures into a single relation may lead to a loss of information, and so, we require specific techniques and formalizations different from the ones commonly applied to single normalized tables. The most basic type that data can exhibit corresponds to the sequential categorical domain, i.e. elements follow in a specific sequential order. These elements in the sequence may have a simple form, such as a single item, or also have a more complex structure, such as sets of items or even a hierarchical organization. This is a complex task due to the combinatorial explosion of searching and generating new patterns, which may range from a plain structure (sequential subsequences) to a more complex tree-like form (such as partial orders).

The sequential mining problem was initially posed by Agrawal and Srikant in [5], and most of the work has focused on providing efficient algorithms for mining frequent patterns of various forms in the sequential data; e.g. works such as [96, 95] are dedicated to the mining of partial orders, and others such as [73, 104, 117, 139] to the mining of frequent subsequences. Recent advances on mining algorithms and probabilistic models for sequential data are surveyed in [45].

In this manuscript, we consider that mining a set of sequences is the first natural step to work towards the closure-based analysis of complex structured objects; the goal here is to provide a theoretical insight into the formalization this domain via formal concept analysis and lattice theory. Intuitions obtained in the sequential case will give a good intuition into other complex combinatorial mining problems, such as having a set of graphs as our input data. This first chapter aims at giving an overview to the specific tasks of mining of sequences that we will be considering through the rest of the book.

1.1 Analysis of Sequences

Let $\mathscr{I} = \{i_1, \ldots, i_m\}$ be a fixed set of items. A subset $I \subseteq \mathscr{I}$ is called an itemset. Formally, we deal with sequential categorical data, described as a collection of ordered transactions $\mathscr{D} = \{d_1, d_2, \ldots, d_n\}$, where each d_i is a *sequence* of finite length. Sequences $d_i \in \mathscr{D}$ will be called input sequences or transactions in this documentation.

We consider a *sequence* to be an ordered list of itemsets. It can be represented as $\langle (I_1)(I_2)\ldots(I_n)\rangle$, where each I_i is a subset of $\mathscr{I}$, and I_i comes before I_j if $i \leq j$. Note that we model each element of the sequence, not as an item, but as an itemset. Without loss of generality we assume that the items in each itemset are sorted in a certain order (such as alphabetic order); and to simplify, itemsets will be displayed without the curly brackets, i.e. ACD represents $\{A, C, D\}$. The universe of all the possible sequences will be denoted by $\mathscr{S}$. Our set of input sequences is a subset of this universe, i.e. $\mathscr{D} \subseteq \mathscr{S}$.

Fig. 1.1 Example of a sequential database $\mathscr{D}$

Seq id	Input sequences
d_1	$\langle(AE)(C)(D)(A)\rangle$
d_2	$\langle(D)(ABE)(F)(BCD)\rangle$
d_3	$\langle(D)(A)(B)(F)\rangle$

This description of $\mathscr{D}$ corresponds exactly to the model of sequences originally proposed by Agrawal and Srikant in [5], and subsequently followed by other works on mining sequential data. This fits exactly in the context of having a *sequential database*, such as a dataset of customer shopping sequences, but it can also fit in case of dealing with *time-series data*, such as alarms in a telecommunication network. In this latter case, the long string of events can be divided into several sliding windows representing each one a piece of $\mathscr{D}$, so that all the transactions would have the same fixed length. A small synthetic example of $\mathscr{D}$ is presented in Figure 1.1.

Formally, we will need some basic operations on sequences.

Definition 1.1 ([5, 117]). We say that a sequence $s = \langle(I_1)\dots(I_n)\rangle$ is a **subsequence** of $s' = \langle(I'_1)\dots(I'_m)\rangle$, i.e. $s \subseteq s'$, if there exist integers $1 \leq j_1 < j_2 \dots < j_n \leq m$ s.t. $I_1 \subseteq I'_{j_1}, \dots, I_n \subseteq I'_{j_n}$; then, we also say that s is **contained** in s'.

For example, the sequence $\langle(C)(D)\rangle$ is contained in $\langle(AC)(D)(B)\rangle$, but it is not contained in $\langle(CD)(A)\rangle$. Reciprocally, we also define that $s \subset s'$ when $s \subseteq s'$ and $s \neq s'$. A sequence is *maximal* in a set of sequences if it is not contained in any other sequence of the set.

Notice that the notion of subsequence that we just introduced is taken directly from the initial work of Agrawal and Srikant in [5, 117], and this has become the commonly accepted formalization. However, we may think of another interpretation of this operation, which is considering equality in the indexes $1 \leq j_1 \leq j_2 \dots \leq j_n \leq m$, thus allowing a sequence to be totally included in one of the itemsets of another sequence. Under this new interpretation a sequence such as $\langle(A)(D)(B)\rangle$ would be included in another such as $\langle(ACD)(B)\rangle$. Formally, this may lead to the disadvantage of letting sequences of infinite length be included into finite ones. For example, an infinite sequence of A's such as e.g. $\langle\dots(A)(A)\dots\rangle$, would be allowed to be included in $\langle(A)\rangle$. This seems an unnatural choice for our applications, where the database $\mathscr{D}$ is composed only of finite input sequences by definition. Here we remain faithful to the interpretation given by Agrawal and Srikant, also to ease the translation of our results w.r.t. former works. We will analyze below the consequences of this choice.

The transaction identifier list of a sequence s w.r.t. $\mathscr{D}$, denoted $tid(s)$, is the list of input sequence identifiers from $\mathscr{D}$ where s is contained, e.g. $tid(\langle(AE)(D)\rangle) = \{d_1, d_2\}$ for the data in Figure 1.1. For short, we will write identifiers with natural numbers, that is, $tid(\langle(AE)(D)\rangle) = \{1,2\}$; later, this simplification of the notation will allow for a comparison with the identifiers used in formal concept analysis. The *support* of a sequence s, denoted as $supp(s)$, is the number of occurrences of s in $\mathscr{D}$; e.g. $supp(\langle(AE)(D)\rangle) = |tid(\langle(AE)(D)\rangle)| = 2$.

Associated to the analysis of sequences there are different tasks and problems. Among all the tasks that one could imagine on such data, we will give a brief overview of the following ones: mining closed sequential patterns, summarizing the data by means of partial orders, mining association rules with order and clustering input sequences.

1.1.1 Mining Closed Sequential Patterns

A relevant task of the sequential mining problem is the identification of frequently-arising patterns or subsequences; in other words, those subsequences in $\mathscr{D}$ whose support is over a user-specified value. These frequent sequential patterns turn out to be useful in many domains, for instance in the anomaly detection for computer security ([85, 88, 89]). Managing sequential patterns and counting their support in $\mathscr{D}$ is a challenging task since one needs to examine a combinatorially explosive number of possible frequent patterns. Many studies have contributed with algorithms for this problem, e.g. [52, 97, 104, 73, 117, 139]. See [45] for a thorough list of references. Unfortunately, there are important cases where the number of frequent patterns is too large for a thorough examination and the algorithms face several computational problems; these include the cases of considering a very low threshold or a dense database (i.e. with high correlation between the items of the input sequences).

Proper solutions to this were initially proposed by Yan, Han and Afshar in [135]. They consists of mining just a compact and more significative set of sequential patterns called the *closed sequential patterns*. This idea parallels the notion of closed itemsets in a binary database, and indeed, both are defined as patterns not extendable to others with the same support. Formally:

Definition 1.2. Given a database $\mathscr{D}$, a sequence $s \in \mathscr{S}$ is **closed** (also known as a **closed sequential pattern**) if there exists no sequence s' with $s \subset s'$ s.t. $supp(s) = supp(s')$.

For instance, taking data from Figure 1.1, we have that $\langle (A)(F) \rangle$ is not closed since it can be extended to $\langle (D)(A)(F) \rangle$ in all the input sequences where it is contained. However, sequences such as $\langle (D)(A) \rangle$ or $\langle (AE)(C) \rangle$ are closed since they are maximal among those others with the same tid list. The set of all the closed sequences and their tid lists from data in Figure 1.1 are presented in Figure 1.2.

For the sake of comparison, in Figure 1.3 we also provide the list of closed sequences that would be derived from the same data in Figure 1.1, yet considering the redefined notion of subsequence that we suggested above. Note that by accepting the total inclusion of one sequence into one itemset we get a more compacted set of final patterns: the classical interpretation of Agrawal and Srikant leads to 9 closed sequences, whereas with the new interpretation we get 6 closed patterns. Despite the potential of this redefinition in practice (under the proper formalizations so as to avoid an infinite number of closed patterns in the set of data), the present document is fully dedicated to the classical interpretation of subsequence.

Fig. 1.2 All closed sequences derived from the data in Figure 1.1

Tid list	Closed Sequential Patterns
$\{1\}$	$\langle (AE)(C)(D)(A) \rangle$
$\{2\}$	$\langle (D)(ABE)(F)(BCD) \rangle$
$\{3\}$	$\langle (D)(A)(B)(F) \rangle$
$\{1,2\}$	$\langle (AE)(C) \rangle$
$\{1,2\}$	$\langle (AE)(D) \rangle$
$\{2,3\}$	$\langle (D)(A)(B) \rangle$
$\{2,3\}$	$\langle (D)(A)(F) \rangle$
$\{2,3\}$	$\langle (D)(B)(F) \rangle$
$\{1,2,3\}$	$\langle (D)(A) \rangle$

Fig. 1.3 All closed sequences derived from data in Figure 1.1 with a new interpretation of the subsequence operation, as explained above

Tid list	Closed Sequential Patterns
$\{1\}$	$\langle (AE)(C)(D)(A) \rangle$
$\{2\}$	$\langle (D)(ABE)(F)(BCD) \rangle$
$\{3\}$	$\langle (D)(A)(B)(F) \rangle$
$\{1,2\}$	$\langle (AE)(C)(D) \rangle$
$\{2,3\}$	$\langle (D)(A)(B)(F) \rangle$
$\{1,2,3\}$	$\langle (D)(A) \rangle$

In [135], Yan, Han and Afshar present the first of a series of algorithms for mining closed sequences in $\mathscr{D}$ over a minimum support, named CloSpan. Later other algorithms followed up to improve its efficiency (e.g. TSP [125] or BIDE [131, 132] or also [38, 109] among many others). The way these algorithms work to identify the closed sequences in the data and their frequency is actually irrelevant for our purposes, and, in general we will use CloSpan as the representative of this group of algorithms that mine closed sequences. Mainly, we consider that the interest in using closures relies on their theoretical characterization: while closed itemsets set up their basis on classical formal concept analysis, there is no such direct formal characterization of the ordered counterpart.

1.1.2 Mining Partial Orders

Alternatively to the mining of frequent sequential patterns, other approaches were designed to go beyond the plain structure of sequential patterns and consider tree-like patterns to summarize the input sequences. The importance of mining more complex structures from sequential data was first argued by Mannila et al. in [96]. The authors consider the mining of frequent episodes, i.e. collections of events occurring frequently together in the input sequences. Episodes are formalized as

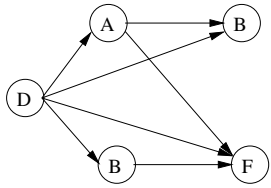

Fig. 1.4 Example of a partial order (also called "hybrid episode")

acyclic directed graphs, and they can be classified into serial episodes (total orders), parallel episodes (trivial orders), and finally, hybrid episodes (general partial orders). In Figure 1.4 there is an example of a general partial order compatible with the second and third input sequences of data in Figure 1.1 (compatibility will be defined formally in chapter 5). Moreover, notice that this partial order is the most specific one for these two input sequences: intuitively, no new node or new edge between the existing nodes can be added to the structure to make it more informative; that is, no other partial order can summarize better those two input sequences.

From the algorithmic perspective, the work in [96] discusses two different approaches for the discovery of frequent episodes in $\mathscr{D}$: Winepi and Minepi. The approach called Winepi is intended to look for frequent episodes following the Apriori scheme, that is, by sliding a window of fixed width along the event sequence. A complete pass along the data is used to compute the support of current episode candidates and, after each pass, new larger episodes are generated as long as the antimonotonicity property of support keeps them active. So, at the end of the process Winepi has discovered all the frequent episodes fitting in the window.

The problem arises with the complexity of managing these structures and the combinatorial explosion to tackle all the cases. Winepi performs two complex operations: first, generating a new set of candidates out of smaller episodes, and second, identifying the compatibility of episodes in each transaction to update the support. In case of mining dense data, and specially, when dealing with hybrid episodes or with noninjective episodes (those where the labels of the nodes can be repeated), the algorithm incurs in a substantial runtime overhead. Apart from this algorithmic overhead, the number of the final discovered episodes is quite large and many of them could be considered redundant: e.g. many of the final parallel episodes may be less informative than some of the serial episodes, and in turn, many serial episodes may be less informative than the hybrid episodes. Yet another objection, if the chosen window is not wide enough, the final discovered episodes will be simply overlapped parts of a longer, more informative episode that cannot fit in the window.

Alternatively to Winepi, the idea of mining unbounded episodes was proposed as a way to solve this latter problem. So, in [96] the same authors present Minepi, which follows again an strategy similar in spirit to Apriori. Another similar attempt presents episodes with gap constraints [53, 98, 79]; or a different idea of frequency that avoids overlapping episodes is presented in [86]. However, these approaches do not fit directly in the transactional model presented here: we are considering input

sequences $d_i \in \mathscr{D}$ of a finite length, and unavoidably, our patterns will be bounded by definition.

Another work worth mentioning is [95] by Mannila and Meek. This method considers a partial order as a generative model for a set of sequences and applies different mixture model techniques. However, they must restrict the attention to a subset of episodes called series-parallel partial orders (such as series-parallel digraphs) to avoid computational problems. Other interesting papers working with episodes are [7] or [41, 124], they deal with serial episodes more than hybrid structures. In general, identifying such hybrid structures directly from the data is a complex task due to the combinatorial nature of the problem.

Other type of probabilistic approaches to identify episode-like models, such as hidden Markov models, are succinctly described in [45]. Finally, the notion of closed episodes has been also studied by several authors, thus proposing efficient algorithms to extend the notion of closed sequential patterns defined above [146, 122, 106, 105]. More formal notions of a closed episode (closed partial order) will be given in chapter 5; our interest is rather to theoretically characterize these notions which are widely used in practice.

1.1.3 Mining Association Rules with Order

A first antecedent to the mining of association rules in sequences is again the work of Mannila et al. in [96]. Actually, we can see serial episodes (total orders) as sequences, and then, an association rule with order has the form $s \to s'$ with $s \subseteq s'$. In other words, we have a sequence implying another sequence, where the antecedent must be contained in the consequent. Of course this idea of implication is based in the classical propositional framework of itemsets, and thus, it can be extended to any hybrid structure, not only sequences.

As it happens with the unordered case, the number of constructed rules can be quite large and difficult to examine. Again it is possible to compute an interestingness measure over the rule with order, such as confidence in [96]. Also other deterministic solutions are naturally possible, e.g. the work in [75] proposes to study just those representative rules obtained through implications of serial closed episodes.

It turns out that serial closed episodes of [75] (which are total orders by definition) are exactly the closed sequences defined in [135] and mined by CloSpan. There is an important algorithmic difference though: the algorithm presented in [75] follows an Apriori scheme, while CloSpan follows the tree structure of PrefixSpan [104]; alternative algorithms have appeared in [125] (TSP) or [131] (BIDE), or recently [38].

In chapter 4 we will consider a generalization of all these association rules by presenting a novel notion of implications where a set of sequences (total orders) in the antecedent imply a single sequence in the consequent. In our case, these sequences in the antecedent may not be necessarily a subsequence of the sequence in the consequent, turning into more informative the predictive rule. Moreover, one

can prove that the set of all these rules exhibits an interesting logical characterization derived from the closure system of sequences.

1.1.4 Clustering Input Sequences

Clustering is the task of grouping together objects into meaningful subclasses. Here transactions in $\mathscr{D}$ can be considered a set of objects described by sequential attributes. The goal is to group objects in $\mathscr{D}$ into different clusters, by using specific discriminating features. This can be useful in different contexts, for example when considering order in the shopping bags of the market basket data, which leads to groups of customers with similar purchasing patterns. One of the key steps in clustering algorithms is the method for computing the similarity between the objects being clustered. The work in [68] by Guralnik and Karypis, considers as discriminating features all the sequential patterns over a certain support with length between two values given by the user. The critical step is then to project the new incoming sequences into the new feature space, so that they must restrict the space to only a reduced set of features. Other notions of clusters for sequences, or representatives of sets of sequences, are based on hidden Markov models, e.g. [116, 87]. In the next chapters we will show that the Galois lattice of closed sequences represents naturally a hierarchical organization of the final clusters.

1.2 Overview of this Book

The main goal of the book is to study the formalization of closed structured patterns in the sequential data. Plain frequent closed sequential patterns proved to be useful in many ways: first, the user needs to examine fewer patterns obtained as an output of the mining algorithms; second, hitting with the right minimum support threshold is not so important, for example, mining all the subsequences in $\mathscr{D}$ with a threshold close to zero is unrealistic and it does not provide useful information, but the set of all closed sequences is not so dramatic and still gives an overall idea of the whole database. In general, our main motivation is that closure systems define a reduced search space with the potential to be characterized by a sound mathematical background based on formal concept analysis. We will completely characterize this closure space of structured patterns for the sequential data, while working at the same time towards the analysis of other structured objects.

First, it is interesting to realize that the set of closed sequential patterns does not represent all the particularities hidden in the sequential data. Formally, there can exist two closed sequences s and s' such that they occur in the same transactions, so that $tid(s) = tid(s')$, but $s \not\subseteq s'$ and $s' \not\subseteq s$. In other words, contrary to the case of closed itemsets in binary data, here there is no unique closure representing a given set of ordered transactions. By way of example, the closed sequences $\langle (AE)(C) \rangle$

and $\langle (AE)(D) \rangle$ from Figure 1.2 occur in the same set of transactions, and none of them can be considered "better" than the other, they simply coexist together.

Indeed, the following research project started with the aim of studying all these particularities of sequential data, by using formal concept analysis as a formalization tool. We rely on the fact that formal concept analysis is a methodology of data analysis and knowledge representation with the potential to be applied to a variety of fields. Indeed, for the unordered context of binary transactions this theory has given already very interesting results, see e.g. [18, 39, 102, 120, 126, 133, 144, 137, 141]. Here, we show that this framework can be very powerful to unify different sequential mining tasks and provide, not only theoretical formulations that help in the understanding, but also new ideas to work towards efficient algorithmic solutions.

As a consequence of the foundations provided for the sequential case, we will show how these contributions can be extended to other kind of structured data which do not contain cycles, mainly represented as partial orders or trees.

Organization

As the simplest case towards the analysis of different structured objects, we take a set of input sequences and we develop a formal framework based on a closure system generated from the foundations of formal concept analysis. This requires to define the proper Galois connection adapted to sequences, and from here, it is direct to construct the concept lattice capturing the particularities and relationships of our data. Then, we use this combinatorial object to formally propose justified methods for the tasks described in this introductory chapter. We organize this document as follows.

Chapter 2 introduces some preliminaries on formal concept analysis for binary data. These foundations will be the first step to understand the characterization of our closure system for sequences in the following chapters.

Chapter 3 sets the mathematical formulation defining our closure system on sequences. This will be the model used as the basis of the subsequent contributions and it corresponds to a lattice of closed sets of sequences. In this chapter we will also prove some basic results that characterize the construction of such a lattice, and the relationship with the classical closed sequential pattern mining. We will provide simple algorithmic schema for the construction of the lattice.

Chapter 4 deals with the problem of defining association rules in ordered data. We will present a novel notion of implication that can be derived from the closure system: a set of sequences imply an individual sequence in the data, and each one of the sequences in the antecedent is not necessarily contained in the sequence of the consequent. We prove that these rules can be formally justified by a purely logical characterization, namely a natural notion of empirical Horn approximation for ordered data which involves specific background Horn conditions. We resolve a way to calculate these implications with order by means of generators of each closed set of sequences in the lattice. The proposed method can be incorporated to current

algorithms for mining closed sequential patterns, of which there are already some in the literature.

Chapter 5 and 6 address the task of summarizing the input sequences by means of partial orders. As a main result we show that the maximal paths of the closed partial orders in the data can also be derived from the nodes of our lattice model. This leads to an interesting algorithmic simplification: we are avoiding the complexity of the mining operation of these structures directly from the input transactions; now we can obtain the hybrid partial orders by just gluing conveniently their maximal paths, which correspond to closed sequential patterns. This result gives the possibility of developing both theoretically and algorithmically the identification of closed partial orders. The work presented in this chapters focuses on the theoretical part of this result, by showing that the identification of a set of maximal paths into a hybrid structure can be characterized with basic operations of category theory [1], through coproducts and colimits. This is not an easy task, specially in the case of considering repeated items in the input sequences. To ease the understanding of the contributions we develop this contribution in two steps. In chapter 5 we simplify the problem to the case of dealing with partial orders where labels cannot be repeated, and in chapter 6 we address the general case of allowing for repeated labels. In chapter 6 we leave as an open question one interesting extension of a result in chapter 5 regarding the property of maximal specificity of our partial order. Assuming this result as a working hypothesis, we can prove the necessary properties that ensure the isomorphy between the closure system of sequences and the corresponding closure system of partial orders.

Chapter 7 develops the extension of the results obtained for the sequential case to other complex structured objects without cycles, which can be very well represented as partial orders. Broadly speaking, each input partial order will be transformed here into a set of sequences corresponding to its maximal paths. Then, a proper notion of subpattern in the new transformed data will allow to directly generate closed structures on the basis of the former theoretical contributions for sequences. Also, this chapter provides the necessary experimental evaluation required to justify our contributions from the more practical point of view.

Chapter 8 concludes and provides a brief overview of the contributions. Most of the results from this book can be found in the following publications [56, 59, 58, 54, 17, 16, 15, 57].