# Mining Sequential Patterns: a Context-Aware Approach

Julien Rabatel, Sandra Bringay and Pascal Poncelet

**Abstract** Traditional sequential patterns do not take into account contextual information associated with sequential data. For instance, when studying purchases of customers in a shop, a sequential pattern could be "*frequently, customers buy products A and B at the same time, and then buy product C*". Such a pattern does not consider the age, the gender or the socio-professional category of customers. However, by taking into account contextual information, a decision expert can adapt his/her strategy according to the type of customers. In this paper, we focus on the analysis of a given context (e.g., a category of customers) by extracting context-dependent sequential patterns within this context. For instance, given the context corresponding to young customers, we propose to mine patterns of the form "*buying products A and B then product C is a general behavior in this population*" or "*buying products B and D is frequent for young customers only*". We formally define such context-dependent sequential patterns and highlight relevant properties that lead to an efficient extraction algorithm. We conduct our experimental evaluation on real-world data and demonstrate performance issues.

Julien Rabatel
Tecnalia, Cap Omega, Rd-Pt B. Franklin, 34960 Montpellier Cedex 2, France,
LIRMM (CNRS UMR 5506), Univ. Montpellier 2, 161 rue Ada, 34095 Montpellier Cedex 5, France, e-mail: rabatel@lirmm.fr

Sandra Bringay
LIRMM (CNRS UMR 5506), Univ. Montpellier 2, 161 rue Ada, 34095 Montpellier Cedex 5, France,
Dpt MIAP, Univ. Montpellier 3, Route de Mende, 34199 Montpellier Cedex 5, France, e-mail: bringay@lirmm.fr

Pascal Poncelet
LIRMM (CNRS UMR 5506), Univ. Montpellier 2, 161 rue Ada, 34095 Montpellier Cedex 5, France, e-mail: poncelet@lirmm.fr

# 1 Introduction

Sequential pattern mining is an important problem widely addressed by the data mining community, with a very large field of applications including the analysis of user behavior, sensor data, DNA arrays, clickstreams, etc. Sequential pattern mining aims at extracting sets of items commonly associated over time. For instance, when studying purchases of customers in a supermarket, a sequential pattern could be *"many customers buy products A and B, then buy product C"*. However, data are very often provided with additional information about purchases, such as the age or the gender of customers. Traditional sequential patterns do not take into account this information. Having a better knowledge about the features of objects supporting a given behavior can help decision making. In this paper, the set of such descriptive information about objects is referred to as contextual information. In the supermarket scenario, the decision expert can adapt his/her strategy by considering the fact that a pattern depends on the type of customer. For instance, an expert who wants to study in detail the *young* customers population could be interested in questions such as "*Are there buying patterns that are frequent for young people, whatever their gender?*" or "*Is there a certain behavior frequent for young people exclusively?*".

Nevertheless, mining such context-dependent sequential patterns is a difficult task. Different contexts should be mined independently to test whether frequent patterns are shared with other contexts. Moreover, some contexts can be more or less general. For instance, the context corresponding to *young* customers is more general than the context corresponding to *young male* customers. Hence, a large number of contexts (more or less general) have to be considered, and the mining process can be very time consuming.

**Related Work.**

Some work in the literature can be seen as related to context-dependent sequential pattern mining. For instance, [Hilderman et al., 1998] define characterized itemsets, i.e., frequent itemsets extracted from a transaction database and associated with values on external attributes defined over a concept hierarchy. Such values can then be generalized using attribute-oriented generalization. Although this notion handles itemsets only, the definitions related to the characterization of patterns could be directly adapted to sequential patterns. However, this work does not take into consideration the representativity of a frequent itemset in a context by considering whether the minimum frequency constraint is satisfied in the sub-contexts. Indeed, an itemset can be associated with a context even if it is not frequent in one or more of its sub-contexts.

Let us consider also the problem of mining multidimensional sequential patterns, i.e., extracting sequential patterns dealing with several dimensions. The first proposition is described in [Pinto et al., 2001], where sequential patterns are extracted in data similar to the contextual data considered in our approach.

However, while multidimensional sequential patterns consider contextual information, they associate a context to a sequential pattern only if this association is globally frequent in the whole database. Moreover, similarly to characterized itemsets, multidimensional sequential patterns do not consider their representativity in their corresponding context. As a consequence, a sequential pattern which is not frequent in the whole database can not be extracted, even if it is very frequent in a sub-category of customers (e.g., the old female customers). We claim in this paper that such a pattern can however be very interesting for a decision maker. Other approaches have considered more complex multidimensional sequence databases, e.g., where items are also described over several dimensions [Plantevit et al., 2005], but the same principles is used to extract such patterns, then leading to the same problems. The same remark can also be mentioned in [Ziembiński, 2007].

To the best of our knowledge, the first approach that tackles this problem for sequential patterns being dependent on more or less general contexts has been proposed in [Rabatel et al., 2010]. However, this work focuses on mining sequential patterns in the whole hierarchy of contexts. Here, we are interested in a different application case: the user focuses on a given context (e.g., young customers) and aims at studying behaviors being related to this context only. We show in this paper that this approach exhibits some interesting properties that can be used in order to efficiently mine context-dependent sequential patterns. In addition, we propose a new type of context-dependent sequential patterns: the exclusive sequential patterns. Intuitively, exclusive sequential patterns are frequent and representative within a context and do not appear frequently elsewhere.

The problem of mining emerging patterns can also be seen as related to context-dependent patterns. Introduced in [Dong and Li, 1999], the mining of emerging patterns aims to extract the itemsets that are discriminant to one class in a set of itemset databases. An emerging pattern is then a pattern whose support is significantly higher in a class than in others. Such patterns can then be exploited to build classifiers [Dong et al., 1999, Li et al., 2001]. For instance, given two data classes $A$ and $B$, emerging patterns in $B$ would be itemsets whose support is significantly higher in $B$ than in $A$. Globally, although emerging patterns and context-dependent sequential patterns (in particular, exclusive sequential patterns) aim at mining patterns that are more frequent in one database than in others, there are some important differences. In particular, the most important problem when mining context-dependent patterns is related to the generalization / specialization order existing amongst contexts. For instance, the context corresponding to young people is more general than the one corresponding to young male people. This aspect is not considered in the mining of emerging patterns, where classes of data do not have such an ordering relation. In addition, our work is only based on the frequency of patterns, while emerging patterns consider a ratio of frequencies over several classes of data.

**Contributions.**

In this paper, we first formally describe contexts. Then, by highlighting relevant properties of such contexts, we show how sequential patterns dependent on one context can be extracted. We conduct experimental evaluation on real-world data and demonstrate performance issues.

More precisely, the following is organized as follows. In Section 2, we define the "traditional" sequential pattern mining problem and show why it is not relevant when contextual information is available. Contextual data, as well as context-dependent sequential patterns, are presented in Section 3. In Section 4, we highlight some relevant properties of context-dependent sequential patterns that are exploited to propose an efficient algorithm. In Section 5, conducted experiments on a real-world dataset are presented. We conclude and discuss future work in Section 6.

## 2 Problem Definition

### *2.1 Traditional Sequential Patterns*

This section describes the traditional sequential pattern mining problem and highlights the need for a specific way to handle contextual information.

Sequential patterns were introduced in [Agrawal and Srikant, 1995] and can be considered as an extension of the concept of frequent itemset [Agrawal et al., 1993] by handling timestamps associated to items. Sequential pattern mining aims at extracting sets of items commonly associated over time. In the *"basket market"* scenario, a sequential pattern could be: *"40 % of the customers buy a television, then buy later a DVD player"*. The problem of mining all sequential patterns in a sequence database is defined as follows.

Let $\mathcal{X}$ be a set of distinct ***items***. An ***itemset*** is a subset of items, denoted by $I = (i_1 i_2 ... i_n)$, i.e., for $1 \leq j \leq n$, $i_j \in \mathcal{X}$. A ***sequence*** is an ordered list of itemsets, denoted by $\langle I_1 I_2 ... I_k \rangle$, where $I_i \subseteq \mathcal{X}$ for $1 \leq i \leq n$.

Let $s = \langle I_1 I_2 ... I_m \rangle$ and $s' = \langle I'_1 I'_2 ... I'_n \rangle$ two sequences. The sequence $s$ is a ***subsequence*** of $s'$, denoted by $s \sqsubseteq s'$, if $\exists i_1, i_2, ... i_m$ with $1 \leq i_1 < i_2 < ... < i_m \leq n$ such that $I_1 \subseteq I'_{i_1}$, $I_2 \subseteq I'_{i_2}$, ..., $I_m \subseteq I'_{i_m}$. If $s \sqsubseteq s'$ we also say that $s'$ ***supports*** $s$.

A ***sequence database*** $\mathcal{D}$ is a relation $\mathcal{R}(ID, S)$, where an element $id \in dom(ID)$ is a sequence identifier, and $dom(S)$ is a set of sequences. The ***size*** of $\mathcal{D}$, denoted by $|\mathcal{D}|$, is the number of tuples in $\mathcal{D}$. A tuple $\prec id, s \succ$ is said to *support* a sequence $\alpha$ if $\alpha$ is a subsequence of $s$, i.e., $\alpha \sqsubseteq s$. The ***support*** of a sequence $\alpha$ in the sequence database $\mathcal{D}$ is the number of tuples in $\mathcal{D}$ supporting $\alpha$, i.e., $sup_{\mathcal{D}}(\alpha) = |\{\prec id, s \succ | (\prec id, s \succ \in \mathcal{D}) \wedge (\alpha \sqsubseteq s)\}|$.

Given a real *minSup* such that $0 < minSup \leq 1$ as the ***minimum support threshold***, a sequence $\alpha$ is ***frequent*** in the sequence database $\mathcal{D}$ if the proportion of tuples in $\mathcal{D}$ supporting $\alpha$ is greater than or equal to *minSup*, i.e., $sup_{\mathcal{D}}(\alpha) \geq minSup \times |\mathcal{D}|$. In this case, sequence $\alpha$ is also called a ***sequential pattern*** in $\mathcal{D}$.

**Table 1** A contextual sequence database.

| id | Age | Gender | Sequence |
|---|---|---|---|
| $s_1$ | young | male | $\langle(ad)(b)\rangle$ |
| $s_2$ | young | male | $\langle(ab)(b)\rangle$ |
| $s_3$ | young | male | $\langle(a)(a)(b)\rangle$ |
| $s_4$ | young | male | $\langle(c)(a)(bc)\rangle$ |
| $s_5$ | young | male | $\langle(d)(ab)(bcd)\rangle$ |
| $s_6$ | young | female | $\langle(b)(a)\rangle$ |
| $s_7$ | young | female | $\langle(a)(b)(a)\rangle$ |
| $s_8$ | young | female | $\langle(d)(a)(bc)\rangle$ |
| $s_9$ | old | male | $\langle(ab)(a)(bd)\rangle$ |
| $s_{10}$ | old | male | $\langle(bcd)\rangle$ |
| $s_{11}$ | old | male | $\langle(bd)(a)\rangle$ |
| $s_{12}$ | old | female | $\langle(e)(bcd)(a)\rangle$ |
| $s_{13}$ | old | female | $\langle(bde)\rangle$ |
| $s_{14}$ | old | female | $\langle(b)(a)(e)\rangle$ |

**Example 1** *Table 1 shows a sequence database describing the purchases of customers in a shop. The first column stands for the identifier of each sequence given in the last column.* $a, b, c, d, e$ *are the products. Column Gender and Age represent extra information about sequences. Such information is not considered in traditional sequential pattern mining. The size of* $\mathcal{D}$ *is* $|\mathcal{D}| = 14$.

*The first sequence in Table 1 describes the sequence of purchases made by a customer identified by* $s_1$*: he has purchased products a and d, then purchased product b.*

*In the following, we set the minimum support minSup to* 0.5. *Let us consider the sequence* $s = \langle(a)(b)\rangle$. *Its support in* $\mathcal{D}$ *is* $sup_{\mathcal{D}}(s) = 8$. *So,* $sup_{\mathcal{D}}(s) \geq minSup \times |\mathcal{D}|$, *thus s is a sequential pattern in* $\mathcal{D}$.

### Why Taking into Account Additional Information?

Considering the previous example, the available contextual information is the age and the gender of customers. A context could be *young female* or *old customer (for any gender)*. Therefore, when considering traditional sequential pattern mining (SPM) on data enriched with contextual information we encounter the following drawbacks.

1. **Some context-dependent behaviors are wrongly considered as general, although they are frequent in only one subcategory of customers**. For instance, $s = \langle (a)(b) \rangle$ is a sequential pattern in $\mathcal{D}$. However, by studying carefully the sequence database, we easily note that $s$ is much more specific to *young* persons. Indeed, 7 out of 8 *young* customers support this sequence, while only 1 out of 6 *old* customers follows this pattern. This problem is directly related to how data cover the customer categories. The fact that young customers are more numerous in the database than old customers allows for a sequence being frequent in young customers only to be frequent in the whole database. However, an expert studying context-dependent patterns in the whole database does not want a pattern being frequent in young customers only to be considered representative in the whole database.

2. **A sequential pattern extracted in a given population does not bring any information about the rest of the population.** For instance, an expert studying frequent behaviors in the *young customers* population will extract the sequence $s = \langle (a)(b) \rangle$. However, the only information provided by this sequential pattern is that *young* customers frequently follow this behavior. An expert can be interested in more information: "*Is this behavior also frequent in the rest of the population or is it exclusively specific to young people?*". Morevover, please note that mining emerging patterns in the young customers population is not a solution here. Indeed, as pointed out for the frequency constraint, if a pattern is emerging in the young customers context compared to the rest of the population, it does not guarantee that it is an emerging pattern for every type of young people.

These drawbacks show that traditional SPM is not relevant when behavior depends on contextual information associated with data sequences. We describe in the following how contextual information are formally handled through mining context-dependent sequential patterns.

## 3 Context-Dependent Sequential Patterns

This section proposes a formal description of contextual data, and defines the different types of context-dependent sequential patterns we aim to mine.

### *3.1 Contextual Sequence Database*

We define a ***contextual sequence database*** $\mathcal{CD}$ as a relation $\mathcal{R}(ID, S, D_1, ...D_n)$, where $dom(S)$ is a set of sequences and $dom(D_i)$ for $1 \leq i \leq n$ is the set of all possible values for $D_i$. $D_1$, $D_2$, ... $D_n$ are called the ***contextual dimensions*** in $\mathcal{CD}$. A *tuple* $u \in \mathcal{CD}$ is denoted by $\prec id, s, d_1, ..., d_n \succ$.

Values on contextual dimensions can be organized as hierarchies. For $1 \leq i \leq n$, $dom(D_i)$ can be extended to $dom'(D_i)$, where $dom(D_i) \subseteq dom'(D_i)$. Let $\subseteq_{D_i}$ be a partial order such that $dom(D_i)$ is the set of minimal elements of $dom'(D_i)$ with respect to $\subseteq_{D_i}$. Then, the partially ordered set $(dom'(D_i), \subseteq_{D_i})$ is the ***hierarchy on dimension*** $D_i$, denoted by $\mathcal{H}_{D_i}$.

**Example 2** *We consider $\mathcal{H}_{Age}$ and $\mathcal{H}_{Gender}$ the hierarchies on dimensions Age and Gender given in Figure 1.*
*In this example, $dom(Age) = \{young, old\}$ and $dom'(Age) = dom(Age) \cup \{*\}$. The partial order $\subseteq_{Age}$ is defined such that young $\subseteq_{Age} *$ and old $\subseteq_{Age} *$.*
*Similarly, $dom(Gender) = \{male, female\}$ and $dom'(Gender) = dom(Gender) \cup \{*\}$. The partial order $\subseteq_{Gender}$ is defined such that male $\subseteq_{Gender} *$ and female $\subseteq_{Gender} *$.*

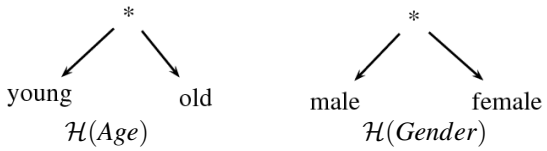


**Fig. 1** Hierarchies on dimensions *Age* and *Gender*.

A ***context*** $c$ in $\mathcal{CD}$ is denoted by $[d_1, ...d_n]$ where $d_i \in dom'(D_i)$. If, for $1 \leq i \leq n$, $d_i \in dom(D_i)$, then $c$ is called a ***minimal context***.

Let $c_1$ and $c_2$ be two contexts in $\mathcal{CD}$, such that $c_1 = [d_1^1, ..., d_n^1]$ and $c_2 = [d_1^2, ...d_n^2]$. Then $c_1 \leq c_2$ iff $\forall i$ with $1 \leq i \leq n$, $d_i^1 \subseteq_{D_i} d_i^2$. Moreover, if $\exists i$ with $1 \leq i \leq n$ such that $d_i^1 \subset_{D_i} d_i^2$, then $c_1 < c_2$. In this case, $c_1$ is said then to be ***more specific*** than $c_2$, and $c_2$ is ***more general*** than $c_1$.
In addition, if $c_1 \not\geq c_2$ and $c_1 \not\leq c_2$, then $c_1$ and $c_2$ are **incomparable**.

**Example 3** *In Table 1, there are four minimal contexts: $[y,m]$, $[y,f]$, $[o,m]$, and $[o,f]$, where y and o respectively stand for young and old, and m and f respectively stand for male and female. In addition, context $[*,*]$ is more general than $[y,*]$ (i.e., $[*,*] > [y,*]$). On the other hand, $[y,*]$ and $[*,m]$ are incomparable.*

The set of all contexts associated with the partial order $\leq$ is called the ***context hierarchy*** and denoted by $\mathcal{H}$. Given two contexts $c_1$ and $c_2$ such that $c_1 > c_2$, $c_1$ is called an ***ancestor*** of $c_2$, and $c_2$ is a ***descendant*** of $c_1$.

For instance, Figure 2 shows a representation of $\mathcal{H}$ for data provided in Table 1 and hierarchies previously given for dimensions *Age* and *Gender*.

Let us now consider the tuples $u = \prec id, s, d_1, ...d_n \succ$ of $\mathcal{CD}$ according to contexts defined above. The context $c = [d_1, ...d_n]$ is called the ***context of*** $u$. Note that the context of $u$ is minimal ($\forall i$ with $1 \leq i \leq n$, $d_i \in dom(D_i)$).

Let $u$ be a tuple in $\mathcal{CD}$ and $c$ the context of $u$. For all contexts $c'$ such that $c' \geq c$ we say that $c'$ ***contains*** $u$ (and $u$ is contained by $c'$).

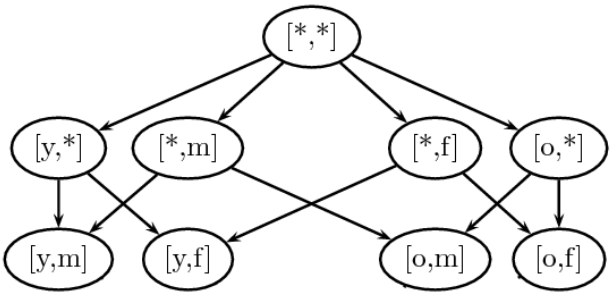


**Fig. 2** The context hierarchy $\mathcal{H}$.

Let $c$ be a context (not necessarily minimal) in $\mathcal{CD}$. The ***sequence database of*** $c$, denoted by $\mathcal{D}(c)$, is the set of tuples contained by $c$. We define the ***size*** of a context $c$, denoted by $|c|$, as the size of its sequence database, i.e., $|c| = |\mathcal{D}(c)|$.

**Example 4** *In Table 1, let us consider contexts* $[o,m]$ *and* $[o,*]$. *Then* $\mathcal{D}([o,m]) = \{s_9, s_{10}, s_{11}\}$ *and* $\mathcal{D}([o,*]) = \{s_9, s_{10}, s_{11}, s_{12}, s_{13}, s_{14}\}$.
*Thus,* $|[o,m]| = 3$ *and* $|[o,*]| = 6$.

## *3.2 Context-Dependent Sequential Patterns*

The previous section showed how a contextual sequence database can be divided into several sequence databases according to contexts.
In the following, we consider the context $c$ and the sequence $s$.

**Definition 1 (c-frequency)** *Sequence s is* ***frequent in*** *c (c-frequent) iff s is frequent in* $\mathcal{D}(c)$, *i.e., if* $sup_{\mathcal{D}(c)}(s) \geq minSup \times |c|$. *We also say that s is a* ***sequential pattern*** *in c. In the following, for simplicity, we note* $sup_{\mathcal{D}(c)}(s)$ *by* $sup_c(s)$.

As seen in Section 2, we focus on mining sequential patterns that, regarding contextual information, are of interest in a given context. We define two different types of patterns that we aim to mine for a given context: general patterns and exclusive patterns.

**Definition 2 (c-generality)** *Sequence s is* ***general in c*** *(c-general) iff:*

1. *s is frequent in c.*
2. *s is frequent in every descendant of c in the context hierarchy.*

The $c$-generality property ensures that a sequential pattern frequent in a given context is also frequent in all its descendants. Hence, such patterns are not sensitive to the problem of data repartition over contexts highlighted in Section 2. For instance, the sequence $\langle(a)(b)\rangle$ that is frequent in the whole database (i.e., the context $[*,*]$) is not general because it is not frequent in the old customers context. Please

note that the set of general sequential patterns in a context is included in the set of sequential patterns.

**Table 2** Sequential patterns in minimal contexts of $\mathcal{CD}$.

| sequence | $[y,m]$ | $[y,f]$ | $[o,m]$ | $[o,f]$ |
|---|---|---|---|---|
| $\langle(a)\rangle$ | **5/5** | **3/3** | **2/3** | **2/3** |
| $\langle(b)\rangle$ | **5/5** | **3/3** | **3/3** | **3/3** |
| $\langle(d)\rangle$ | 2/5 | 1/3 | **3/3** | **2/3** |
| $\langle(e)\rangle$ | 0/5 | 0/3 | 0/3 | **3/3** |
| $\langle(a)(b)\rangle$ | **5/5** | **2/3** | 1/3 | 0/3 |
| $\langle(b)(a)\rangle$ | 0/5 | **2/3** | **2/3** | **2/3** |
| $\langle(bd)\rangle$ | 1/5 | 0/3 | **3/3** | **2/3** |

**Example 5** *Table 2 shows, from the contextual sequence database provided in Table 1, the sequences being frequent in at least one minimal context as well as their support for each minimal context (of the form $sup_c(s)/|\mathcal{D}(c)|$). When the support is displayed in bold, then the sequence is frequent in the corresponding minimal context.*

*Let us now consider the context $[o,*]$ (corresponding to old people). According to Definition 2, a sequence s is general in the context $[o,*]$ iff s is frequent in $[o,*]$ (i.e., the context itself), $[o,m]$ and $[o,f]$ (i.e., its descendants in the context hierarchy).*

*All sequences $\langle(a)\rangle$, $\langle(b)\rangle$, $\langle(d)\rangle$, $\langle(b)(a)\rangle$ and $\langle(bd)\rangle$ satisfy these conditions. They are $[o,*]$-general. On another hand, sequence $\langle(e)\rangle$ is frequent in $[o,*]$ (it is supported by 3 old customers of 6) but not in its descendant $[o,m]$. In consequence, $\langle(e)\rangle$ is not general in $[o,*]$.*

However, *c*-generality only considers whether a given sequential pattern in a context is frequent in its descendants, without considering the rest of the context hierarchy. We therefore propose *c*-exclusive sequential patterns, by considering whether there exists another context in the rest of the hierarchy (except *c* and its descendants) where *s* is general.

**Definition 3 (c-exclusivity)** *Sequence s is **exclusive in c** (c-exclusive) iff:*

1. *s is general in c.*
2. *there does not exist a context $c' \not\leq c$ such that s is $c'$-general.*

In other words, a *c*-exclusive sequential pattern is general in *c* and *c*'s descendants only. That can be seen as a discriminance constraint w.r.t. the *c*-generality property.

**Example 6** *According to definition 3, a sequence s is exclusive in the context $[o,*]$ iff s is general in $[o,*]$, $[o,m]$ and $[o,f]$ only. Given the sequences being general in $[o,*]$, we can note that only two of them meet this requirement: $\langle(d)\rangle$ and $\langle(bd)\rangle$. On another hand, sequence $\langle(b)(a)\rangle$ is also general in $[*,f]$ and therefore is not exclusive in $[o,*]$.*

We have defined general and exclusive sequential patterns, two types of context-dependent sequential patterns. Those patterns are frequent sequences in a context $c$ that satisfy some interesting properties regarding contextual information. Such sequences can be exploited to assist a user in a context-driven analysis of data.

# 4 Mining Context-Dependent Sequential Patterns

In this section, we detail the properties that will help us to efficiently mine context-dependent patterns defined in previous section.

## *4.1 Preliminary definitions and properties.*

In the following, we will mainly rely on the minimal contexts of the hierarchy. In order to easily manipulate these elements, we define the decomposition of a context as the set of its minimal descendants.

**Definition 4 (decomposition of a context)** *Let $c$ be a context. The decomposition of $c$ in $\mathcal{CD}$, denoted by $decomp(c)$, is the non-empty set of minimal contexts such that $\forall c' \in decomp(c)$, $c \geq c'$.*

**Example 7** *The decomposition of context $[y,*]$ is $\{[y,m],[y,f]\}$.*

The decomposition of a context $c$ forms a partition of its sequence database. Consequently, we can highlight some immediate set-theoretical properties.

**Lemma 1.** *Let $c$ be a context, $decomp(c) = \{c_1, c_2, ..., c_n\}$ its decomposition and $s$ a sequence. Given the definition of the sequence database of $c$, the decomposition of $c$ has the following properties:*

1. $\bigcap_{i=1}^{n} \mathcal{D}(c_i) = \emptyset$;
2. $\bigcup_{i=1}^{n} \mathcal{D}(c_i) = \mathcal{D}(c)$;
3. $|c| = |\mathcal{D}(c)| = \sum_{i=1}^{n} |c_i|$;
4. $sup_c(s) = \sum_{i=1}^{n} sup_{c_i}(s)$.

Lemma 1 can be exploited to unveil an interesting property about the $c$-frequency of sequential patterns in the decomposition of $c$.

**Lemma 2.** *Let $c$ be a context, and $decomp(c) = \{c_1, c_2, ..., c_n\}$. If $\forall i \in \{1, ..., n\}$, $s$ is frequent in $c_i$, then $s$ is frequent in $c$. In addition, $s$ is frequent in all the descendants of $c$.*

***Proof:*** *For each $c_i$ such that $i \in \{1,..n\}$, $sup_{c_i}(s) \geq minSup \times |c_i|$. This means $\sum_{i=1}^{k} sup_{c_i}(s) \geq \sum_{i=1}^{n} minSup \times |c_i|$. However, $\sum_{i=1}^{n} minSup \times |c_i| = minSup \times \sum_{i=1}^{n} |c_i| = minSup \times |c|$. Since $\sum_{i=1}^{k} sup_{c_i}(s) = sup_c(s)$ it follows $sup_c(s) \geq minSup \times |c|$.*

*Let $c'$ be a context such that $c > c'$. Then $decomp(c') \subseteq decomp(c)$, i.e., $s$ is frequent in all contexts in $decomp(c')$. Applying the previous result we obtain $s$, a frequent sequence in $c'$.* □

In the following, we show how we benefit from such properties in order to efficiently mine context-dependent patterns.

We first have interest in mining $c$-general sequential patterns. Given the definition of a $c$-general sequential pattern, a naive approach could be performed in the following steps:

1. Mine sequential patterns in $c$.
2. Mine sequential patterns in every descendants of $c$.
3. Output sequential patterns frequent in $c$ and all its descendants.

However, the number of contexts to mine can be very large (i.e., $c$ and $c$'s descendants). In order to overcome this drawback, we exploit the properties of the context hierarchy and show that $c$-general sequential patterns can be extracted by focusing only on the decomposition of $c$.

**Theorem 1** *The sequence $s$ is $c$-general iff $\forall c' \in decomp(c)$, $s$ is frequent in $c'$.*

***Proof:*** *If $s$ is frequent in each context of $decomp(c)$, then, by applying Lemma 2, $s$ is frequent in $c$ and $c$'s descendants in the context hierarchy, i.e., it is general in $c$. In addition, if $\exists c' \in decomp(c)$ such that $s$ is not frequent in $c'$, then there exists a descendant of $c$ where $s$ is not frequent and $s$ is not general in $c$ according to Definition 2.* □

Theorem 1 is a key result as it guarantees that $c$-general sequential patterns are sequences being frequent in all minimal descendants of $c$. This property can therefore be exploited in order to mine general sequential patterns in a context $c$. Moreover, $c$-generality provides us with the following lemma.

**Lemma 3.** *If a sequence $s$ is not $c$-general, then $\forall s'$ such that $s' \sqsupseteq s$, $s'$ is not $c$-general.*

***Proof:*** *If $s$ is not $c$-general, then there exists a context $c' \leq c$ where $s$ is not frequent. Given $s'$ a sequence such that $s' \sqsupseteq s$, $s'$ is also not frequent in $c'$. As a result, $s'$ is not $c$-general.* □

Lemma 3 shows that $c$-generality is anti-monotonic with respect to the size of the sequence. This property will be useful when coming to extract $c$-general sequential patterns.

We also aim at mining $c$-exclusive sequential patterns. A naive approach to mine such patterns could be done in the following steps:

1. Mine general sequential patterns in $c$ (see previous naive approach).
2. Mine general sequential patterns in each context $c'$ of the hierarchy that is not $c$ or a descendant of $c$.
3. Output general sequential patterns in $c$ that are not general in any other context $c'$.

This approach is very time consuming, as it requires to mine sequential patterns in all contexts of the hierarchy. However, a similar reasoning as for $c$-general sequential patterns can be applied to redefine the $c$-exclusivity property.

**Lemma 4.** *There exists a context $c' \not\leq c$ such that $s$ is $c'$-general iff there exists a minimal context $c''$ such that $c'' \notin decomp(c)$ and $s$ is frequent in $c''$.*

***Proof:*** *If $s$ is $c'$-general, then $s$ is frequent in each element of $decomp(c')$. However, if $c' \not\leq c$, then at least one element of $decomp(c')$ is not an element of $decomp(c)$. So, there exists a minimal context $c''$ such that $c'' \notin decomp(c)$ and $s$ is frequent in $c''$.*

*Moreover, if there exists a minimal context $c''$ such that $c'' \notin decomp(c)$ and $s$ is frequent in $c''$, then $s$ is $c''$-general. However, $c'' \not\leq c$. As a result, there exists a context $c' \not\leq c$ such that $s$ is $c'$-general.* □

**Theorem 2** *Let $\mathcal{M}$ be the set of minimal contexts in $\mathcal{H}$. The sequence $s$ is $c$-exclusive iff:*

*1. $\forall c' \in decomp(c)$, $s$ is frequent in $c'$,*
*2. $\forall c'' \in \mathcal{M} \setminus decomp(c)$, $s$ is not frequent in $c''$.*

***Proof:*** *This result is obtained by directly applying Theorem 1 and Lemma 4 to the definition of a c-exclusive sequential pattern (Definition 3).* □

Hence, a $c$-exclusive sequential pattern is a $c$-general sequential pattern $s$ such that there does not exist a minimal context outside the decomposition of $c$ where $s$ is frequent.

Hence, Theorems 1 and 2 show that both general and exclusive sequential patterns can be mined by considering minimal contexts only, while naive approaches require to consider all descendants of $c$ to extract $c$-general sequential patterns, and all the contexts of the hierarchy to mine $c$-exclusive sequential patterns. In the following, we propose an algorithm that exploits these results in order to efficiently mine context-dependent sequential patterns.

## *4.2 Algorithm*

This section presents *Gespan*, an algorithm designed to mine both general and exclusive sequential patterns in a given context. It is based on the *PrefixSpan* algorithm

[Pei et al., 2004] that aims at solving traditional sequential pattern mining. We explain the principles of *PrefixSpan* in the following example, by describing the process of mining sequential patterns in the sequence database $\mathcal{D}$ from Table 1, with a minimum support threshold set to *0.5*.

**Example 8** *A scan of the sequence database extracts all the sequential patterns of the form $\langle(i)\rangle$, where i is an item. Hence, PrefixSpan finds $\langle(a)\rangle$, $\langle(b)\rangle$, $\langle(d)\rangle$, since $\langle(c)\rangle$ and $\langle(e)\rangle$ are not frequent.*

*In consequence, the whole set of sequential patterns in $\mathcal{D}$ can be partitioned into subsets, each subset being the set of sequential patterns having $\langle(i)\rangle$ as a prefix. These subsets can be extracted by mining the* ***projected databases*** *for each prefix, i.e., for each $\langle(i)\rangle$. A projected database contains, for each data sequence, its subsequence containing all frequent items following the first occurrence of the given prefix. Such a subsequence is called a* ***postfix****. If the first item x of the postfix is in the same itemset as the last item of the prefix, the postfix is denoted by $\langle(\_x\ldots)\ldots\rangle$.*

*Then, $\langle(a)\rangle$ is outputted, and the $\langle(a)\rangle$-projected database is built, containing 11 postfixes: $\langle(\_d)(b)\rangle$, $\langle(\_b)(b)\rangle$, $\langle(a)(b)\rangle$, $\langle(bc)\rangle$, etc. Then items i, such that either $\langle(ai)\rangle$ or $\langle(a)(i)\rangle$ is frequent, are extracted from the $\langle(a)\rangle$-projected database. b is such an item, as $\langle(a)(b)\rangle$ is a sequential pattern. So, the process continues by outputting $\langle(a)(b)\rangle$, and using it as a new prefix.*

We now present the *Gespan* algorithm that aims at mining general and exclusive sequential patterns in a context.

The prefix-growth approach of PrefixSpan is used to extract general sequential patterns, relying on the anti-monotonicity of the *c*-generality property. From a prefix sequence *s*, the algorithm builds the *s*-projected database by making use of the method *BuildProjectedDatabase*, and scans the projected database (method *ScanDB*) to find items *i* that can be assembled to form a new general sequential pattern $s'$. Then, the $s'$-projected database is built and the process continues. Since the general intuition is similar to *PrefixSpan*, we do not detail the *ScanDB* and *BuildProjectedDatabase* methods of *Gespan*, but only focus on the differences.

In method $ScanDB(\mathcal{CD})$, the support of *i* is computed in each minimal context of the *s*-projected database. Testing the *c*-generality of the resulting sequence $s'$ in the projected database is based on Theorem 1 and performed by method $isGeneral(s', C, \mathcal{H})$ described in Algorithm 2.

Then, for each *c*-general sequential pattern, method $isExclusive(s', C, \mathcal{H})$ described in Algorithm 3 is used to test whether this sequential pattern is also *c*-exclusive. Please note that if the user is only interested in mining *c*-general sequential patterns, this step of the algorithm can be removed.

## 5 Experiments

All experiments have been performed on a system equipped with a 3GHz CPU and 16GB of main memory. The methods are implemented in C++.

**Algorithm 1** Gespan

**Input:** $\mathcal{CD}$ a contextual sequence database, *minSup* a minimum support threshold, $\mathcal{H}$ a context hierarchy, $C$ a context in $\mathcal{H}$.
Call $subGespan(\langle\rangle, \mathcal{CD}, \mathcal{H}, C)$;

**Subroutine** subGespan($s, \mathcal{CD}, \mathcal{H}, C$)
**Input:** $s = \langle I_1 ... I_n \rangle$ a sequence; $\mathcal{CD}$ the $s$-projected database, $\mathcal{H}$ a context hierarchy, $C$ a context in $\mathcal{H}$.
$ScanDB(\mathcal{CD})$
Let $\mathcal{I}$ be the set of items $i$ such that $isGeneral(\langle I_1 ... (I_n \cup i) \rangle, C, \mathcal{H})$ returns $TRUE$
Let $\mathcal{I}'$ be the set of items $i$ such that $isGeneral(\langle I_1 ... I_n (i) \rangle, C, \mathcal{H})$ returns $TRUE$
**for all** $i \in (\mathcal{I} \cup \mathcal{I}')$ **do**
  $s'$ is the sequence such that $i$ is appended to $s$
  **if** isExclusive($s'$, $C$, $\mathcal{H}$) **then**
    output $s'$ as a $C$-exclusive sequential pattern
  **end if**
  output $s'$ as a $C$-general sequential pattern
  $\mathcal{CD}' = BuildProjectedDatabase(s', \mathcal{CD})$
  call $subGespan(s', \mathcal{CD}', \mathcal{H})$
**end for**

**Algorithm 2** isGeneral($s$, $C$, $\mathcal{H}$)

**Input:** $s$ a sequence, $C$ a context, $\mathcal{H}$ a context hierarchy.
**for all** $c \in decomp(C)$ **do**
  **if** $s$ is not frequent in $c$ **then**
    **return** FALSE
  **end if**
**end for**
**return** TRUE

**Algorithm 3** isExclusive($s$, $C$, $\mathcal{H}$)

**Input:** $s$ a pattern, $C$ a context, $\mathcal{H}$ a context hierarchy.
Let $\mathcal{M}$ be the set of minimal contexts in $\mathcal{H}$
**for all** $c \in \mathcal{M} \setminus decomp(C)$ **do**
  **if** $s$ is frequent in $c$ **then**
    **return** FALSE
  **end if**
**end for**
**return** TRUE

By conducting these experiments, we wish to evaluate the performances of *Gespan* by focusing on two aspects.

**Number of patterns.** We study the number of context-dependent sequential patterns extracted with *Gespan*, and compare it to the number of frequent sequences (i.e., sequential patterns) extracted with *PrefixSpan*. Indeed, we show in Section 2 that some traditional sequential patterns are irrelevant when considering the analysis of contextual data. This experiment will allow to quantify this aspect.

**Runtime.** We measure the execution time required to mine context-dependent patterns in a given context, and compare it to the time required to mine frequent sequences in the same context.

## *5.1 Data Description*

The experiments were conducted on about 100000 product reviews from *amazon.com*, in order to study the vocabulary used according to reviews. This dataset is a subset of the one used in [Jindal and Liu, 2008]. Reviews have been lemmatized[1] and grammatically filtered in order to remove uninteresting terms, by using the *tree tagger* tool [Schmid, 1994]. Preserved terms are verbs (apart from modal verbs and the verb *"to be"*), nouns, adjectives and adverbs. Remaining terms have been stemmed[2] using the Porter algorithm [Porter, 1980]. Then, the sequence database is constructed using the following principles:

- each review is a sequence,
- each sentence is an itemset (i.e., the order of the words in a sentence is not considered),
- each word is an item.

An extracted sequential pattern could be $\langle(eat\ mushroom)(hospital)\rangle$, which means that frequently a review contains *eat* and *mushroom* in a sentence and *hospital* in one of the following sentences.

### Contextual dimensions.

Each review is associated with contextual dimensions:

- the *product* type (*Books*, *DVD*, *Music* or *Video*)
- the *rating* (originally a numeric value $r$ between 0 and 5). For these experiments, $r$ has been translated into qualitative values: *bad* (if $0 \leq r < 2$), *neutral* (if $2 \leq r \leq 3$), and *good* (if $3 < r \leq 5$)
- the proportion of helpful *feedbacks*[3], i.e., *0-25%*, *25-50%*, *50-75%* or *75-100%*.

We define hierarchies on contextual dimensions as described in Figure 3. The number of contexts in the context hierarchy is $|dom'(product)| \times |dom'(rating)| \times |dom'(feedbacks)| = 6 \times 5 \times 7 = 210$, while the number of minimal contexts is $|dom(product)| \times |dom(rating)| \times |dom(feedbacks)| = 4 \times 3 \times 4 = 48$.

---

[1] i.e., the different forms of a word have been grouped together as a single item. For instance, the different forms of the verb *to be* (is, are, was, being, etc.) are all returned as *to be*.

[2] i.e., the inflected forms of a word are reduced to their root form. For instance, the adjective *musical* is returned as *music*.

[3] On amazon.com each reader can post a feedback on a review.

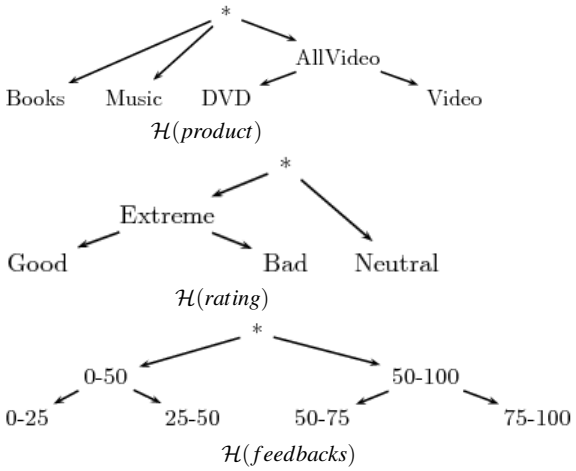


**Fig. 3** Hierarchies on contextual dimensions.

Note that the domain of values of contextual dimensions has been enriched with new values. For instance, hierarchy $\mathcal{H}(rating)$ contains an element *Extreme* that will allow us, for instance, to extract patterns being general in extreme opinions (positive or negative).

## *5.2 Results and Discussion*

It is not possible to show the obtained results for each of the 210 contexts in the hierarchy. As a consequence, we will provide the results for a selection of more or less general contexts:

- $[*,*,*]$ is the more general context of the hierarchy. It corresponds to all the reviews in the database.
- $[Books,*,*]$ is the context corresponding to all the reviews that are related to a book.
- $[Books,bad,*]$ is a more specific context than $[Books,*,*]$. It corresponds to bad reviews associated to a book.
- $[Books,bad,75-100]$ is a minimal context in the hierarchy. It corresponds to bad reviews associated to a book that have been considered useful by more than 75% of voting amazon users.

Table 3 presents the number of patterns extracted with $minSup = 0.01$ for each algorithm: *PrefixSpan* to extract sequential patterns (i.e., frequent sequences), *Gespan* to extract general sequential patterns (GSP) and exclusive sequential patterns (ESP). First, please note that the number of general sequential patterns is significantly lower than the number of sequential patterns in all non-minimal contexts. This shows that

**Table 3** Number of sequential patterns, general sequential patterns and exclusive sequential patterns, according to the context, for $minSup = 0.01$.

| Context | PrefixSpan | Gespan (GSP) | Gespan (ESP) |
|---|---|---|---|
| $[*,*,*]$ | 50089 | 1788 | 1788 |
| $[Books,*,*]$ | 79113 | 15147 | 5179 |
| $[Books,bad,*]$ | 194765 | 62661 | 603 |
| $[Books,bad,75-100]$ | 259790 | 259790 | 19801 |

a large proportion of patterns that are frequent in a context are actually specific to a sub-part of this context only. However, the number of general sequential patterns in $[Books,bad,75-100]$ is equal to the number of sequential patterns. Indeed, because minimal contexts have no descendants, a sequential pattern in a minimal context is general in this context.

Second, the number of exclusive sequential patterns is significantly lower than the number of general sequential patterns in all contexts, except for $[*,*,*]$. For instance, only 1% of general sequential patterns in $[Books,bad,*]$ is actually exclusive in this context. However, the number of exclusive sequential patterns in $[*,*,*]$ is equal to the number of general sequential patterns. Indeed, there does not exist a context that is not a descendant of $[*,*,*]$. As a result, all general sequential patterns in $[*,*,*]$ are also exclusive.

These results are directly related to the definition of frequent, general and exclusive sequential patterns. General sequential patterns are indeed frequent sequential patterns that satisfy the representativity constraint required by the $c$-generality. As a consequence, the set of general patterns in a context is included in the set of frequent patterns. Similarly, exclusive sequential patterns are general sequential patterns in a context satisfying an additional constraint (being general in this context and its sub-contexts only). The set of exclusive patterns in a context is therefore included in the set of general patterns of the same context.

**Table 4** Runtime in seconds for extracting each type of sequential patterns, according to the context, for $minSup = 0.01$.

| Context | PrefixSpan | Gespan (GSP) | Gespan (GSP + ESP) |
|---|---|---|---|
| $[*,*,*]$ | 1795 | 131 | 131 |
| $[Books,*,*]$ | 1435 | 463 | 646 |
| $[Books,bad,*]$ | 449 | 216 | 1344 |
| $[Books,bad,75-100]$ | 212 | 212 | 2477 |

Table 4 shows the execution time needed to mine each type of sequential patterns. Two versions of *Gespan* have been used. The first one aims at mining general sequential patterns only, while the second aims at mining both general and exclusive sequential patterns. Mining general sequential patterns in non-minimal contexts is

always faster than mining sequential patterns. We also note that the gap in the runtime is larger when the considered context is more general.

The time required to extract exclusive sequential patterns strongly depends on the level of generalization of the mined context. Indeed, when the considered context is very general (e.g., $[*,*,*]$ or $[Books,*,*]$) then mining exclusive sequential patterns is faster than mining sequential patterns. However, for more specific contexts (e.g., $[Books,bad,*]$ or $[Books,bad,75-100]$) the mining of exclusive sequential patterns is time consuming. This is due to the number of minimal contexts that must be considered in order to test the exclusivity of a sequential pattern. For instance, in order to test whether a general sequential pattern $s$ is exclusive in $[Books,bad,75-100]$, *Gespan* needs to check whether $s$ is frequent in one of the 47 other minimal contexts in the hierarchy. This number of minimal contexts is lower when the considered context is more general. Hence, mining exclusive sequential pattern is more efficient in more general contexts.

In addition, please note that we have not compared *Gespan* with the baseline approaches described in Section 4, but only with *PrefixSpan*. The reason is that baseline approaches are very naive, and obviously more time-consuming than *PrefixSpan*. Moreover, the comparison with *PrefixSpan* allows us to confront the two advantages of *Gespan* over a traditional sequential pattern mining algorithm. First, general or exclusive sequential patterns are more informative than frequent sequential patterns, as they consider only representative patterns when contextual information is available. Second, *Gespan* exploits theoretical properties highlighted in Section 4 and offers reduced runtimes (except for mining exclusive patterns in very specific contexts, as shown in Table 4).

## 6 Conclusion

In this paper we have motivated the need for mining context-dependent sequential patterns in a sequence database enriched with contextual information. We formally defined the problem and unveiled set-theoretical properties that allow database mining in a concise manner.

This work can be extended in a number of ways. First, in this paper we have specifically handled sequential patterns. An immediate prospect is the generalization of this work to other types of frequent patterns such as frequent episodes [Mannila et al., 1997] or frequent subgraphs [Kuramochi and Karypis, 2001]. Second, we have only focused on a minimum support threshold to extract context-dependent sequential patterns. In future work, we aim at studying other constraints. For instance, we have already pointed out in Section 1 that mining general and exclusive patterns can be seen as related to the problem of mining emerging patterns. An interesting prospect consists in mining context-dependent emerging patterns by adapting the corresponding constraint to the notion of $c$-generality and $c$-exclusivity defined for context-dependent sequential patterns.

## References

[Agrawal et al., 1993] Agrawal, R., Imieliński, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. *SIGMOD Rec.*, 22(2).

[Agrawal and Srikant, 1995] Agrawal, R. and Srikant, R. (1995). Mining sequential patterns. In Yu, P. S. and Chen, A. S. P., editors, *Eleventh International Conference on Data Engineering*. IEEE Computer Society Press.

[Dong and Li, 1999] Dong, G. and Li, J. (1999). Efficient mining of emerging patterns: discovering trends and differences. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA. ACM.

[Dong et al., 1999] Dong, G., Zhang, X., Wong, L., and Li, J. (1999). CAEP: Classification by Aggregating Emerging Patterns. In *Discovery science: second international conference, DS'99, Tokyo, Japan, December 6-8, 1999, proceedings*, page 30. Springer Verlag.

[Hilderman et al., 1998] Hilderman, R., Carter, C., Hamilton, H., and Cercone, N. (1998). Mining market basket data using share measures and characterized itemsets. *Research and Development in Knowledge Discovery and Data Mining*, pages 159–173.

[Jindal and Liu, 2008] Jindal, N. and Liu, B. (2008). Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining*. ACM.

[Kuramochi and Karypis, 2001] Kuramochi, M. and Karypis, G. (2001). Frequent subgraph discovery. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, pages 313–320. IEEE.

[Li et al., 2001] Li, J., Dong, G., and Ramamohanarao, K. (2001). Making use of the most expressive jumping emerging patterns for classification. *Knowledge and Information systems*, 3(2):131–145.

[Mannila et al., 1997] Mannila, H., Toivonen, H., and Inkeri Verkamo, A. (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1(3):259–289.

[Pei et al., 2004] Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U., and Hsu, M. (2004). Mining sequential patterns by pattern-growth: the PrefixSpan approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(11).

[Pinto et al., 2001] Pinto, H., Han, J., Pei, J., Wang, K., Chen, Q., and Dayal, U. (2001). Multi-dimensional sequential pattern mining. In *Proceedings of the tenth international conference on Information and knowledge management*. ACM.

[Plantevit et al., 2005] Plantevit, M., Choong, Y. W., Laurent, A., Laurent, D., and Teisseire, M. (2005). $M^2SP$: Mining sequential patterns among several dimensions. In Jorge, A., Torgo, L., Brazdil, P., Camacho, R., and Gama, J., editors, *PKDD*, volume 3721 of *Lecture Notes in Computer Science*. Springer.

[Porter, 1980] Porter, M. (1980). An algorithm for suffix stripping. *Program: Electronic Library & Information Systems*, 40(3):211–218.

[Rabatel et al., 2010] Rabatel, J., Bringay, S., and Poncelet, P. (2010). Contextual Sequential Pattern Mining. In *2010 IEEE International Conference on Data Mining Workshops*, pages 981–988. IEEE.

[Schmid, 1994] Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, volume 12. Citeseer.

[Ziembiński, 2007] Ziembiński, R. (2007). Algorithms for context based sequential pattern mining. *Fundamenta Informaticae*, 76(4):495–510.