# C3Ro: An efficient mining algorithm of extende d-close d contiguous robust sequential patterns in noisy data 

Y Abboud, Armelle Brun, Anne Boyer

## To cite this version:

Y Abboud, Armelle Brun, Anne Boyer. C3Ro: An efficient mining algorithm of extende d-close d contiguous robust sequential patterns in noisy data. Expert Systems with Applications, Elsevier, 2019, 131, pp.172-189. 10.1016/j.eswa.2019.04.058 . hal-02977461

## HAL Id: hal-02977461 <br> https://hal.inria.fr/hal-02977461

Submitted on 25 Oct 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# C3Ro: an Efficient Mining Algorithm of Extended-Closed Contiguous Robust Sequential Patterns in Noisy Data 

Y. Abboud, A. Brun, A. Boyer<br>Université de Lorraine, LORIA UMR 7503, France


#### Abstract

Sequential pattern mining has been the focus of many works, but still faces a tough challenge in the mining of large databases for both efficiency and apprehensibility of its resulting set. To overcome these issues, the most promising direction taken by the literature relies on the use of constraints, including the well-known closedness constraint. However, such a mining is not resistant to noise in data, a characteristic of most real-world data. The main research question raised in this paper is thus: how to efficiently mine an apprehensible set of sequential patterns from noisy data? In order to address this research question, we introduce 1) two original constraints designed for the mining of noisy data: the robustness and the extended-closedness constraints, 2) a generic pattern mining algorithm, C3Ro, designed to mine a wide range of sequential patterns, going from closed or maximal contiguous sequential patterns to closed or maximal regular sequential patterns. C3Ro is dedicated to practitioners and is able to manage their multiple constraints. C3Ro also is the first sequential pattern mining algorithm to be as generic and parameterizable. Extensive experiments have been conducted and reveal the high efficiency of C3Ro, especially in large datasets, over well-known algorithms from the literature. Additional experiments have been conducted on a real-world job offers noisy dataset, with the goal to mine activities. This experiment offers a more thorough insight into C3Ro algorithm: job market experts confirm that the constraints we introduced actually have a significant positive impact on the apprehensibility of the set of mined activities.


Keywords: Data mining, Sequential pattern mining, Closed contiguous pattern, Noisy data, Constraints, Efficiency

## 1. Introduction

Pattern mining [5] is one of the most studied topics in the data mining literature. A sequential pattern is a pattern whose order of items is considered and applications rely on their mining: pattern discovery in protein sequences [59, 36], analysis of customer behavior in web logs [10, 11], sequence-based classification [4], etc. Consequently, sequential pattern mining [6] has become a significant part of pattern mining studies. Many sequential pattern mining algorithms have been proposed, such as regular sequential pattern mining [46, 26, 27, $56,7,13]$, string mining [12, 41], closed sequential pattern mining [42, 43, 57, 51, 47, 23, 13], constraint-based sequential pattern mining [22, 44, 8, 32], etc., which all

[^0]have substantially improved the domain.
Although large databases are becoming quite common, mining sequential patterns in such data is still a computationally expensive task, when not an intractable one. In addition, the complete set of sequential patterns is often difficult to apprehend for the final user, due to its large size and to the presence of useless or redundant patterns. This is especially limiting when the final user is a practitioner who expects a set of patterns that fits his/her requirements. Therefore, the domain still faces tough challenges related to both

- the efficiency of the mining algorithms
- the apprehensibility of the resulting set of patterns. We propose to define the efficiency of an algorithm as the ability to run in an optimal execution time and memory usage; and the apprehensibility of a set of patterns as a trade-off between, on one side its size and redundancy in its patterns, and on the other side the information contained in the set.

To overcome both issues, the most promising direction adopted by the literature relies on the use of constraints on the patterns that results in a reduced set of patterns [44, 32, 24]. We identify two types of constraints: one applies to a set of sequential patterns, and the other one applies to the intrinsic characteristics of each sequential pattern.

The first type of constraints includes the closedness $[42,43,57,51,47]$ that preserves all the information, the generator-pattern [19, 53, 14, 33] and the maximal-pattern $[46,39,21,31,16]$ constraints that reduce the size of the set of patterns at the cost of a part of the information. Closedness has become popular as the resulting set of patterns is less redundant, which increases its apprehensibility. However, despite this improvement [57, 48, 51, 23, 13], the set remains unmanageable in large databases [58] and in noisy data $[52,38,54,37]$ as most of the benefit of closedness is lost. Noise, which is a characteristic inherent to many real-world applications [20], is the random appearance of disturbances in data, such as loss, substitution, or addition of items.
With the double objective to mine a more apprehensible set and to increase the efficiency of the mining algorithm, we introduce a new constraint: the extendedclosedness constraint. This new constraint relaxes the constraint of identical values required by the closedness constraint. Thus, the set of extended-closed sequential patterns can be viewed as a trade-off between the set of closed sequential patterns, which contains all the information, and the set of maximal sequential patterns, which is an even more reduced set, but that contains a subset of the information.

The second type of constraints includes a large number of constraints such as monotonic or anti-monotonic constraints [44], regular expression constraints [22], gap constraints [34], contiguity constraint [8, 58, 2] (a gap constraint with a value of 0 ), etc. The contiguity constraint is the most popular one. Many real-life tasks greatly benefit from this constraint: text mining [18], Web log mining [10, 9], DNA and amino acid sequences mining [30, 31], etc. The contiguity constraint leads to the extraction of much fewer patterns, with a shorter average length, and with almost no loss of information [58], resulting in a more apprehensible set, while improving the efficiency of the algorithm. However, the added value provided by the contiguity constraint faces limitations in case of noisy data. Indeed, the support of the contiguous patterns becomes unreliable [38], making some them being frequent or
at the opposite being not frequent. The set of patterns is thus unreliable too. Wildcards are a way to manage noisy data [37], even with a contiguity goal [2]. A wildcard $[34,49,50,37]$ is a joker of one item, regarding the contiguity constraint. However, using wildcards in contiguous sequential pattern mining may result in the mining of semi-contiguous sequential patterns, which decreases the apprehensibility of the resulting set.
To make the contiguity constraint noise-resistant and thus the set of patterns more apprehensible, we introduce a second constraint: the robustness constraint. A robust sequential pattern is a frequent contiguous sequential pattern that occurs a limited number of times with wildcards.

Each constraint from the literature allows to solve a specific issue in sequential pattern mining. However, practitioners' final needs are often so specific that they correspond to a combination of such constraints. For example, contiguous sequential patterns, in noisy data, which contain a particular set of items. To date, in order to obtain the set that fits the desired combination of constraints, the use of several algorithms is often required. However, there is no guarantee of compatibility, consistency or scalability of the entire process [8]. Combining several algorithms and solving their possible compatibility issues can be inaccessible to many practitioners which limits their usability.
To alleviate this issue, we introduce C3Ro, a simple and generic algorithm. The key idea of C3Ro is to integrate several constraints and the associated parameters, to mine a wide range of sequential patterns, going from closed or maximal contiguous sequential patterns to closed or maximal sequential patterns, including semi-contiguous sequential patterns. In addition, it is noise-resistant thanks to the use of wildcards and of the extended-closedness constraint. It thus increases the efficiency and the apprehensibility of the set of mined patterns.
Hence, the main contributions of this paper are:

- we introduce two new constraints: robustness and extended-closedness, designed to improve both the efficiency of the mining process and the apprehensibility of a set of closed contiguous sequential patterns in noisy data. These constraints have many practical applications in different fields, especially those related to humans who are often a source of noisy data. Apprehensibility is specifically thought for practitioners, with the goal to mine the smallest set that contains the highest amount of information;
- we introduce C3Ro, a generic and highly param- 185 eterizable algorithm that can be used in various 186 contexts and adaptable to many user needs. It is ${ }_{187}$ especially designed for practitioners to help them ${ }_{188}$ mine patterns using a single tool, while considering all their requirements. To reach this gener- 190 icity, C3Ro manages a set of monotonic or antimonotonic constraints, a number of wildcards, robustness and extended ratios.

This paper is organized as follows. In section 2 we discuss the related work. In Section 3 we introduce concepts and notations. Section 4 defines the two new constraints. The C3Ro algorithm is introduced in Section 5. Section 6 is dedicated to the experiments conducted, including a real-world dataset in the domain of job market and Section 7 concludes this work and discusses some perspectives.

## 2. Related Work

The sequential pattern mining problem was first introduced by Agrawal and Srikant in [6], with the Apriori ${ }^{207}$ algorithm. Apriori is based on the monotonic constraint ${ }^{208}$ of the frequency: "all nonempty subsets of a frequent 209 itemset must also be frequent" [5]. Since then, many se- 210 quential pattern mining algorithms have been proposed, ${ }_{211}$ still based on the monotonic constraint, with the goal of ${ }_{212}$ improving the mining efficiency in terms of execution ${ }^{213}$ time and memory usage, such as GSP [46], PrefixS- 214 pan [27], SPADE [56], SPAM [7], CM-SPADE [13], 215 CM-SPAM [13], etc. The PrefixSpan algorithm uses a 216 pattern-growth philosophy through projected databases ${ }_{217}$ to mine frequent sequential patterns. PrefixSpan recur- 218 sively extends a prefix sequential pattern by adding a 219 frequent item from the projected database of this prefix. 220 The SPADE algorithm, released the same year, is based ${ }_{221}$ on a vertical IDLists representation and uses a lattice- ${ }^{222}$ theoretic approach to decompose the original search ${ }^{223}$ space into smaller spaces. One year after, the SPAM ${ }^{224}$ algorithm used a different representation, it is a vertical ${ }^{225}$ bitmap representation dedicated to the mining of long ${ }^{226}$ sequential patterns. While being faster than SPADE and ${ }^{227}$ PrefixSpan for the mining of long patterns, SPAM uses ${ }^{228}$ a large amount of memory. The newer CM-SPADE and ${ }^{229}$ CM-SPAM algorithms introduced a co-occurrence ma- 230 trix to substantially reduce the number of join opera- ${ }^{231}$ tions between the IDLists of the candidate sequential ${ }^{232}$ patterns and thus improve the efficiency of the mining. ${ }_{233}$ Although all these algorithms contributed to the im- ${ }^{234}$ provement of the mining process, they still face tough ${ }^{235}$
challenges in the mining efficiency and in the apprehensibility of the results. Several directions are used in the literature to overcome those issues. The two main directions are the use of a compact representation of sequential patterns [25] and the reduction of the number of sequential patterns mined [46]. In this paper, we focus on the latter and the most promising way reach this goal is through the use of constraints [46, 22, 44, 8]. Using constraints during the mining process allows to extract less patterns, and improves the efficiency. However, the usage of constraints improves the apprehensibility if the discarded patterns only contain information irrelevant to the practitioners. Therefore, we are only interested in constraints that preserve relevant information, i.e., that are able to adapt to the application context of the practitioners. To facilitate the analysis of the constraints from the literature, we propose to classify them in two categories that we name: (1) Global constraints, i.e., constraints that relate to the characteristics of a set of sequential patterns. (2) Local constraints, i.e., constraints that relate to the intrinsic characteristics of each sequential pattern.

### 2.1. Global constraints

Global constraints [43, 39, 19] are, for example, the well-known maximal, generator and closedness constraints. With Global constraints, the reduction of set of frequent patterns may be reached at the cost of a part of the information. However, it is impossible to control both the amount and the nature of the information lost, this loss can thus be prejudicial for many applications. The maximal constraint $[21,16]$ mines only the frequent sequential patterns that are not included in any other frequent sequential pattern. Therefore, the set of maximal sequential patterns is significantly smaller than the complete set of sequential patterns. However, the information (especially the support value) of the sequential patterns included in maximal patterns is lost, which often represents a non-negligible part of the information.
The generator constraint [53, 14, 33] mines sequential patterns that do not contain any other sequential pattern with the same support. The set of generator sequential patterns is also significantly smaller than the set of sequential patterns. The information loss is about all the sequential patterns that contain the generator sequential patterns, especially about those that are not frequent.
The closedness constraint [57] mines sequential patterns that have no super-pattern with the same support. The set of closed sequential patterns is often larger than both maximal and generator sequential patterns sets, and remains smaller than the set of sequential patterns. Moreover, the closedness constraint does
not cause any loss of information. Closedness is by far ${ }^{287}$ the most popular Global constraint and numerous algo- ${ }^{288}$ rithms $[43,51,34,23,13,33,45]$ focus on the mining ${ }_{289}$ of closed patterns.
The Clospan [51] algorithm is one of the first algorithms ${ }^{291}$ dedicated to the mining of closed sequential patterns. 292 Clospan mines and stores a set of candidate patterns, 293 based on PrefixSpan, then a post processing phase filters ${ }_{294}$ out the non closed ones. Storing the set of candidates 295 is expensive in terms of memory. To cope with this ${ }_{296}$ issue, the BIDE algorithm [47] proposes to not mem- ${ }^{297}$ orize any candidate pattern. BIDE is also based on 298 PrefixSpan, but uses a BI-Directional closedness check- 299 ing scheme to not generate any set of candidates, hence 300 BIDE improves greatly both execution time and mem- 301 ory usage. More recent algorithms such as Clasp [23] 302 and CloFast [17] adopt a vertical IDLists representation ${ }^{303}$ to outperform Clospan and BIDE in terms of execution 304 time, but at the expense of a higher memory usage due ${ }_{305}$ to the candidate generation phase. CM-Clasp [13] and 306 FCloSM [33] algorithms both use a co-occurrence ma- 307 trix to reduce the number of candidates and improve both execution time and memory usage.
A remaining major issue in the mining of closed sequential patterns lies in the low apprehensibility of the set of mined patterns in large databases. This is especially true for vertical IDLists-based algorithms, which are not ${ }_{3}$ efficient due to their high memory usage. The patterngrowth philosophy appears to be more suited. However, ${ }^{314}$ the issue gets worse when mining noisy data [2].

In a nutshell, on one hand, the maximal and the gen- ${ }^{316}$ erator constraints allow to significantly reduce the set 317 of mined patterns. On the other hand, these constraints ${ }_{318}$ also cause a significant and unpredictable loss of infor- 319 mation, which is often inacceptable for practitioners. 320 The closedness constraint seems more relevant as no 321 information is lost. However, its issue with large and ${ }^{322}$ noisy databases needs to be addressed.

### 2.2. Local constraints

Local constraints [46, 40,55] regroup numerous constraints such as monotonic or anti-monotonic con- ${ }^{327}$ straints: presence of an item or a pattern, maximum ${ }^{328}$ or minimum length, duration, etc. [44], regular ex- ${ }^{329}$ pression constraints [22], gap constraints [34], includ- 330 ing contiguity constraint, etc. Local constraints are ${ }_{331}$ practitioner-dependent: they are oriented towards the ${ }_{332}$ application context, according to the practitioner in- ${ }^{33}$ terests. One of the very first sequential pattern min- ${ }^{334}$ ing algorithm that has integrated Local constraints is ${ }_{335}$ GSP [46]. GSP allows to use duration and gap con- ${ }^{336}$ straints on all the sequential patterns that it extracts.

These two constraints are widely used in the literature [29, 28, 15]. The SPIRIT algorithm is the first algorithm that integrates regular expression constraints [22], it converts the constraints into an automaton that prunes the patterns during the mining. The monotonic and antimonotonic constraints allow the pattern-growth philosophy to identify in an early stage if the patterns formed, when extending a prefix, will satisfy or not a set of constraints [44]. For this reason, several algorithms like PG [44], CloSPEC [8], global-p.f [32], CCPM [2], based on the pattern-growth philosophy, are able to mine patterns satisfying an aggregate of monotonic or anti-monotonic constraints.
A commonly used gap constraint is the contiguity constraint (gap $=0$ ) $[18,10]$, especially used for natural language processing or document classification. Recently, [58] has shown that the information contained in the set of contiguous sequential patterns is almost equivalent to the information contained in the set of sequential patterns. The contiguity constraint is therefore a Local constraint that can be used to reduce the number of mined patterns. Consequently, the use of the contiguity constraint improves both the efficiency of the mining and the apprehensibility of the results. However, the mining of contiguous patterns has the drawback of being unable to mine noisy data, without a significant loss of information.

### 2.3. Combination of constraints

Both Global and Local constraints are used to reduce the number of sequential patterns mined, while retaining either the complete information, of at least the most important information for the practitioner. In addition to improving the apprehensibility of the results, these constraints also improve the efficiency of the algorithm since the decrease in the number of sequential patterns mined has a very favorable impact on execution time and memory usage [47, 44]. Therefore, both the ability of algorithms to integrate some constraints and the possibility for the practitioner to specify them, are critical.
Several algorithms allow to use multiple constraints during the mining. For instance, the PG algorithm [44] uses rules to integrate a set of monotonic and antimonotonic constraints during the mining of sequential patterns. Many algorithms like CloSPEC [8] and CCPM [2] use such rules in addition to the closedness constraint. Unfortunately, although Local constraints are often effective to improve the efficiency of the mining process and the apprehensibility of its result, they also depend on the application context of the practitioner and thus are not always specifiable.
Recently the CCSpan [58] algorithm introduced closed
contiguous sequential pattern ( $\mathrm{ClCoS} P$ ) mining. Fre- 387 quent $\mathrm{ClCoS} P$ are far fewer and shorter than frequent ${ }_{388}$ closed sequential patterns, while containing the same ${ }_{389}$ information [58] than closed sequential patterns. CC- 390 Span outperforms several closed sequential pattern min- 391 ing algorithms like Clospan and BIDE, but scales rather ${ }^{392}$ poorly on datasets with long sequences [2]. The CCPM ${ }_{393}$ algorithm [2] also mines $\operatorname{ClCoS} P$ based on the same ${ }_{394}$ pattern-growth philosophy than BIDE and PG. CCPM ${ }_{395}$ outperforms and scales better than CCSpan, while al- ${ }^{396}$ lowing to use in addition monotonic or anti-monotonic ${ }_{397}$ constraints. However, although $\operatorname{ClCoSP}$ mining is an ${ }_{398}$ effective combination of constraints to reduce the size 399 of the resulting set of patterns, it has the noise resistance 400 limit of the contiguity constraint.

### 2.4. Pattern mining in noisy data

As many real-world applications generate noisy or uncertain data, more and more works focus on the mining of sequential patterns in such data [ $3,38,54,35,37]$. In these works, data is qualified as either noisy data [38, 54] or uncertain data [37, 34]. The common approach adopted by these works notices that it is impossible to count the frequency of patterns deterministically, so do adopt a probabilistic approach. In addition, they assume statistical independence between the different items in terms of their uncertain probability behavior. Thus they evaluate the probabilistic support of each itemset or pattern in each sequence to determine if the appearance of an itemset in a pattern is likely to be an error or not.

The approach proposed in [38] adopts a constraintbased approach, the model proposed places constraints on the fraction of errors permitted in each item column and the fraction of errors permitted in a supporting transaction. Taken together, these constraints make the patterns that contain systematic errors being discarded.

The major drawback of the probability-based approach is its high computational complexity, it is thus not efficient and does not meet our requirements.

### 2.5. Summary

In summary, the literature highlights that a wide range of sequential pattern mining algorithms mine a (too) high number of patterns. This set is not ${ }^{428}$ apprehensible. Two main directions are adopted to 429 solve this issue. The first direction aims at decreasing ${ }_{430}$ the set of patterns while maximizing the amount of ${ }_{431}$ information, without depending on the application context of the practitioner. Like the well-known closed sequential pattern mining or contiguous sequential pattern mining. However, none of them is able to
manage large and noisy databases, which is common in many real-world applications [38]. The second direction aims at decreasing the set of patterns while extracting only the information relevant to the practitioner, by using Local constraints. A local constraint is practitioner-dependent, and significantly improves the apprehensibility and the efficiency of the mining. Although many algorithms (BIDE, PG, CCPM) based on the pattern-growth philosophy allow the combination of several constraints, no noise-resistant algorithm aggregates as many constraints as monotonic, anti-monotonic, closedness and contiguity constraints. In other words, most algorithms lack genericity.
As a consequence, there is a critical need for a generic algorithm able to manage large and noisy databases.

The issues raised by the state-of-the art make us consider the following questions:

1) How to efficiently mine an apprehensible set of sequential patterns, including in the frame of noisy data?
2) How to design a sequential pattern mining algorithm that aggregates several constraints such as monotonic, anti-monotonic, closedness and contiguity constraints and thus can be used with several combinations of constraints, whatever are the needs of practitioners are?

## 3. Preliminaries

In this section, we introduce some definitions and notations that will be further used in this paper.
Let $I=\left\{i_{1}, i_{2}, \ldots, i_{n}\right\}$ be a set of distinct items. An itemset $I_{m} \subseteq I$ with $\left|I_{m}\right|$ being the length of $I_{m}$, is an unordered set of distinct items. Without loss of generality, we assume that items in itemsets are sorted according to a total order, such as the lexicographic order to facilitate the reading and for further processing. A sequential pattern $P$ is an ordered list of itemsets, denoted by $P=\left\langle E_{1}, E_{2}, \ldots, E_{j}\right\rangle$ where $E_{k} \subseteq I$ with $1 \leq k \leq n$. In this paper, an itemset is always written with an uppercase letter, while an item is always written with a lowercase letter.

Let $e$ be an item of $I, P \diamond e$ means $P$ concatenates with $e$. Concatenation can be an I-extension (I for item), $P \diamond_{i} e=\left\langle E_{1}, E_{2}, \ldots, E_{m} \cup\{e\}\right\rangle$ or an Sextension (S for sequence), $P \diamond_{s} e=\left\langle E_{1}, E_{2}, \ldots, E_{m},\{e\}\right\rangle$. We expand the definition of item extension to sequential pattern extension. Given $P=\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$ and $P^{\prime}=\left\langle E_{1}^{\prime}, E_{2}^{\prime}, \ldots, E_{m}^{\prime}\right\rangle$ two sequential patterns, $P \diamond P^{\prime}$ means $P$ concatenates with $P^{\prime}$. When it is a I-extension, $P \diamond_{i} P^{\prime}=\left\langle E_{1}, E_{2}, \ldots, E_{n} \cup\left\langle E_{1}^{\prime}, E_{2}^{\prime}, \ldots, E_{m}^{\prime}\right\rangle\right\rangle$ and when it is a S-extension, $P \diamond_{s} P^{\prime}=\left\langle E_{1}, E_{2}, \ldots, E_{n}, E_{1}^{\prime}, E_{2}^{\prime}, \ldots, E_{m}^{\prime}\right\rangle$.

An input sequence database $S D B$ is a set of tuples ${ }_{484}$ (sid, $S$ ), where sid is a sequence id, and $S$ an input se- ${ }^{485}$ quence. A sequence can also be considered as a pattern, thus it shares the same properties. The number of tuples in $S D B$ is called the size of $S D B$ and is denoted by $|S D B|$. A wildcard is a symbol that matches any itemset of the sequence database. Aside from the above notations, we further present some definitions.

Definition 3.1 (Contiguous sub-pattern). Given $P=$ $\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$ and $P^{\prime}=\left\langle E_{1}^{\prime}, E_{2}^{\prime}, \ldots, E_{m}^{\prime}\right\rangle$ two sequential patterns, $P$ is a contiguous sub-pattern of $P^{\prime}$, denoted as $P \sqsubseteq P^{\prime}$, if and only if $n \leq m$ and there exist consecutive integers $j_{1}, j_{2}, \ldots, j_{n}$ such that: (1) $1 \leq j_{1}<j_{2}<\ldots<$ $j_{n} \leq m$; and (2) $E_{1} \subseteq E_{j_{1}}^{\prime}, E_{2} \subseteq E_{j_{2}}^{\prime}, \ldots, E_{n} \subseteq E_{j_{n}}^{\prime} . P^{\prime}$ is called a contiguous super-pattern of $P$, and $P^{\prime}$ is said to contain $P$.

Definition 3.2 (Absolute, Relative and Universal support). The absolute support of a sequential pattern $P$ in a sequence database $S D B$ is the number of tuples in $S D B$ that contain $P$, denoted by $\sup _{A}^{S D B}(P)$. The relative support of $P$ in $S D B$ is the proportion of tuples in $S D B$ that contain $P$, denoted by $\sup _{R}^{S D B}(P)$. The universal support of $P$ in $S D B$ is the number of occurrences of $P$ in $S D B$, denoted by $\sup _{U}^{S D B}(P)$.

Without loss of generality, we use the absolute support for describing the C3Ro algorithm and use the rel- 5 ative support to present the experimental results in the remaining of the paper.

Definition 3.3 (Frequent closed contiguous sequential pattern). Given min_sup a support threshold, a contiguous sequential pattern $P$ is frequent in $S D B$ if $\sup _{A}^{S D B}(P) \geq$ min_sup. $P$ is a frequent closed contiguous sequential pattern if there exists no sequential pattern $P^{\prime}$ such that: $P \sqsubset P^{\prime}$, and $\sup _{A}^{S D B}(P)=$ $s u p_{A}^{S D B}\left(P^{\prime}\right)$.

We now present some definitions related to the contiguous projected items of a prefix sequential pattern, in the context of contiguous sequential pattern mining. Given an input database $S D B$, an input sequence ${ }^{5}$ $S=\left\langle E_{1}^{\prime}, E_{2}^{\prime}, \ldots, E_{m}^{\prime}\right\rangle$ and a sequential pattern $P={ }^{521}$ $\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$ with $P \sqsubseteq S$. Therefore, there exist con- ${ }^{522}$ secutive integers $j_{1}, j_{2}, \ldots, j_{n}$ such that: (1) $1 \leq j_{1}<$ $j_{2}<\ldots<j_{n} \leq m ;$ and (2) $E_{1} \subseteq E_{j_{1}}^{\prime}, E_{2} \subseteq E_{j_{2}}^{\prime}, \ldots, E_{n} \subseteq$ $E_{j_{n}}^{\prime}:$
Definitio 3.4 (Contiguous projecteditems of a sequen- 527 tial pattern). The contiguous projected items of $P$ in $S{ }_{528}$ are the union of two sets: (1) the set of i-extensions $e_{i}$ of ${ }_{529}$ each occurrence of $P$ in $S$ such that $E_{n} \diamond_{i} e_{i} \subseteq E_{j_{n}}^{\prime}$, (2) if 530
$m>n$, the set of s-extensions $e_{s}$ of each occurrence of $P$ in $S$ such that $e_{s} \in E_{j_{n+1}}^{\prime}$.

Definition 3.5 (Contiguous projected sequence). The set of contiguous projected items of $P$ in $S$ is called the contiguous projected sequence of $P$ in $S$.

Definition 3.6 (Prefix contiguous sequential pattern). $P$ is called the prefix contiguous sequential pattern.

Definition 3.7 (Contiguous projected database). The set of contiguous projected sequences of $P$ in $S D B$ is called the contiguous projected database of $P$ in $S D B$, denoted $P_{-c} S D B$

## 4. Introduction of two New Constraints: Robustness and Extended-Closedness

### 4.1. ClCoS P mining in noisy data

ClCoS $P$ mining is currently one of the most efficient way to mine sequential patterns in data [58, 2]. In our review of the literature, we have raised the issue of mining closed contiguous sequential patterns in noisy data. As mentioned in the previous section, the noise in data represents a part of data that does not bring any additional information. In practice, the noise in data corresponds to three types of disturbances: loss, substitution or addition of items in the data (notice that most of the works of the literature only consider addition of items). The noise is therefore a random disturbance, poorly reproducible and infrequent, generally not found in frequent patterns. However, the noise decreases the support of some patterns, and can even make them not being part of the resulting set of patterns. For example, in a text mining context, noise may correspond to word omission (loss), spelling errors (substitution) or the use of irrelevant words (addition). The sentence "Elementary, my dear Watson" can thus become, in noisy data, "Elementary, my Watson" or "Elementary, my deer Watson" or even "Elementary, my dear old Watson" in case of imperfect typing of a user. However, this is in fact, three times the same pattern "Elementary, my dear Watson" with different types of noise. Let us set up a closed contiguous sequential pattern mining with a minimum support min_sup $=2$ in the sequence database Noise (presented in Table 1) made up of 5 sequences: two occurrences are the sequence〈Elementary, my, dear, Watson〉 without noise and three occurrences occur with noise.
With the purpose of illustrating the impact of noise in closed contiguous sequential pattern mining, we exceptionally look for a specific pattern in the resulting

Table 1：Example of noisy sequence database Noise

| sid | sequence | ClCoSP |
| :--- | :--- | :--- |
| $\# 1$ | Elementary my dear Watson | $\langle$ Elementary，my $\rangle: 5,\langle$ Elementary，my，dear $\rangle: 3$, |
| $\# 2$ | Elementary my Watson | $\langle$ Elementary，my，dear，Watson $\rangle: 2$, |
| $\# 3$ | Elementary my deer Watson | $\langle$ Watson $\rangle: 5$ |
| $\# 4$ | Elementary my dear old Watson |  |
| $\# 5$ | Elementary my dear Watson |  |

set．Indeed，the pattern 〈Elementary，my，dear，Watson〉 573 represents the information we are looking for．In the 574 context of sequential pattern mining，we define the 575 information contained in a pattern as the tuple（pattern，${ }^{576}$ support of the pattern）．
From this mining，the pattern ${ }^{578}$〈Elementary，my，dear，Watson〉 has a support of 2579 in this base of five sequences．This low frequency is 580 partly due to the low support of the item dear：3，which is substituted once（＂deer＂）and omitted once．The low support of the item dear causes a split of the pattern：${ }^{5}$〈Elementary，my，dear，Watson〉 in two more frequent ${ }^{583}$ sub－patterns：〈Elementary，my $\langle$ ，〈Watson $\rangle$ ．The set of ${ }^{584}$ closed contiguous sequential patterns extracted then ${ }^{585}$ lacks irreversibly part of the information．In order to make $\operatorname{ClCoS} P$ relevant when mining noisy data，the three types of disturbances have to be handled．All the conclusions that will be drawn on the closed contiguous sequential pattern mining $\operatorname{ClCoS} P$ remain valid in contiguous sequential pattern mining $\operatorname{CoS} P$ ．
On the one hand，wildcards $[34,49,50]$ can be used to address the addition of items issue［2］．On the other hand，their use may lead to the mining of patterns that may not be relevant in the frame of contiguous sequen－ tial pattern mining．Indeed，a contiguous sequential pat－ tern that always occurs with wildcards is unlikely to be a contiguous sequential pattern that occurs with＂noise＂ items．It is more likely that it is a semi－contiguous sequential pattern，i．e．a contiguous sequential pat－ tern whose all occurrences use a limited number of wildcards．For example，in the Noise database，the pattern 〈Elementary，my，dear，Watson〉 occurs twice contiguously（\＃1，\＃5）and once with one wildcard （\＃4）．It can thus be considered as a contiguous pattern． However，the pattern 〈Elementary，my，Watson〉 occurs contiguously only once（\＃2）and four times with one or two wildcards（\＃1，\＃3，\＃4，\＃5）．This pattern is more likely to be a semi－contiguous pattern．

We now introduce some definitions related to the us－ age of wildcards in $\operatorname{CoS} P$ mining．Let $k \in \mathbb{N}$ be the number of wildcards authorized．

Definition 4.1 （k－contiguous sub－pattern）．Given $P=613$
$\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$ and $P^{\prime}=\left\langle E_{1}^{\prime}, E_{2}^{\prime}, \ldots, E_{m}^{\prime}\right\rangle$ two sequential patterns，$P$ is a $k$－contiguous sub－pattern of $P^{\prime}$ ，denoted as $P \sqsubseteq_{k} P^{\prime}$ ，if and only if $n<m$ and there exist integers $j_{1}, j_{2}, \ldots, j_{n}$ such that：（1） $1 \leq j_{1}<j_{2}<\ldots<j_{n}<m$ ；and（2）$E_{1} \subseteq E_{j_{1}}^{\prime}, E_{2} \subseteq E_{j_{2}}^{\prime}, \ldots, E_{n} \subseteq E_{j_{n}}^{\prime}$ ；and（3） $j_{n}-j_{1}-n \leq k . P^{\prime}$ is called a $k$－contiguous super－pattern of $P$ ，and $P^{\prime} k$－contains $P\left(j_{n}-j_{1}-n\right.$ is the number of wildcards actually used）．

Definition 4.2 （Absolute，Relative，Universal and Wild－ card $k$－support）．The absolute $k$－support of a sequen－ tial pattern $P$ in a sequence database $S D B$ is the num－ ber of tuples in $S D B$ that $k$－contain $P$ ，denoted by $\sup _{A W_{k}}^{S D B}(P)$ ．The relative $k$－support of $P$ in $S D B$ is the proportion of sequences in $S D B$ that $k$－contain $P$ ，de－ noted by $\sup _{R W_{k}}^{S D B}(P)$ ．The universal $k$－support of $P$ in $S D B$ is the number of occurrences with $k$ wildcards of $P$ in $S D B$ ，denoted by $\sup _{U W_{k}}^{S D B}(P)$ ．The wildcard $k$－support of a sequential pattern $P$ in a sequence database $S D B$ is the number of tuples that $k$－contain $P$ using at least 1 wildcard，denoted by $\sup _{W_{k}}^{S D B}(P)$ ．

The subtraction of the wildcard $k$－support from the $k$－support of a pattern makes it possible to obtain the number of sequences where the pattern appears strictly respecting the contiguity constraint．
We now present in Table 2 the mining of closed 1－ contiguous sequential patterns（with $k=1$ wildcard）on the previous basis（Table 1），specifying the 1 －support and wildcard 1 －support of each pattern．Patterns im－ pacted by the use of a wildcard are in marked red．The purpose of this new mining is still to extract the pattern〈Elementary，my，dear，Watson〉，which represents the information．We see that using $k=1$ wildcard allows to extract a new pattern：〈Elementary，my，Watson〉． This pattern does not contain all the information we are looking for，this is explained by the fact that this pattern is mined three times out of four times thanks to the use of a wildcard．Therefore，in the context of a closed contiguous sequential pattern mining，we consider the pattern 〈Elementary，my，Watson〉 as not being a contiguous pattern．
As expected，the wildcard allows to

Table 2：Use of one wildcard in the mining of $\operatorname{ClCoS} P$ in the Noise database

| sid | sequence | ClCoS $P$ with one wildcard：support（wildcard 1－support） |
| :--- | :--- | :--- |
| $\# 1$ | Elementary my dear Watson | $\langle$ Elementary，my $\rangle: 5(0),\langle$ Elementary，my，dear，Watson $\rangle: 3(1)$, |
| $\# 2$ | Elementary my Watson | $\langle$ Elementary，my，Watson $\rangle: 4(3)$, |
| $\# 3$ | Elementary my deer Watson | $\langle$ Watson $\rangle: 5(0)$ |
| $\# 4$ | Elementary my dear old Watson |  |
| $\# 5$ | Elementary my dear Watson |  |

find additional occurrences of the pattern 652〈Elementary，my，dear，Watson〉，here one addi－${ }^{653}$ tional occurrence in the sequence \＃4 and so allows 654 to overcome the presence of the＂noise＂item old． 655 This increase of 1 of the 1 －support allows the pattern ${ }_{656}$〈Elementary，my，dear，Watson〉 to have the same 1－support as the pattern 〈Elementary，my，dear〉，${ }_{658}$ and as $\left\langle\right.$ Elementary，my，dear $\quad \sqsubseteq_{1}$〈Elementary，my，dear，Watson〉，it is no longer ${ }^{659}$ closed and therefore is removed from the resulting set．
In conclusion，the use of $k=1$ wildcard makes it possible to alleviate the appearance of additional items and thus to extract information not mined so far．In 661 return，the use of wildcards mines patterns that do not ${ }_{662}$ necessarily respect the contiguity constraint．These ${ }_{663}$ patterns are therefore not relevant for the extraction 664 of information in the frame of closed contiguous 665 sequential pattern mining．The problem is thus to limit 6 the use of wildcards．

Definition 4.3 （ $k$－contiguous $\delta$－robust sequential pat－${ }_{680}$ tern）．Given $\delta \in[0 ; 1]$ a ratio and $S D B$ a sequence ${ }_{681}$ database，a $k$－contiguous sequential pattern $P$ is $\delta$－robust if and only if $\frac{\sup _{W_{k}}^{S D B}(P)}{\sup _{A W_{k}}^{S D B}(P)} \leq \delta$ ．

This definition reflects the fact that $k$－contiguous 686 $\delta$－robust sequential patterns stand on the proportion of ${ }_{687}$ contiguous occurrences of a pattern against occurrences ${ }^{688}$ that use wildcards．This proportion is set by the value 689 of the robustness ratio $\delta$ ，which is fixed by the user， 690
depending on the estimated amount of noise in the data．The mining of $k$－contiguous $\delta$－robust sequential patterns，thus relies on the use of wildcards and a robustness ratio（see next section）．

In our running example（Table 1），for both patterns mined with $k=1$ wildcard： $\frac{s u p_{W_{k}}^{\text {Noise }}(\langle\text { Elementary，my，} \text { Watson }\rangle)}{\text { sup }_{A W_{k}}^{\text {Noise }}(\langle\text { Elementary，my，} \text { Watson }\rangle)}=\frac{3}{4}=0.75 \mathrm{et}$ $\frac{\sup _{W_{k}}^{\text {Noise }}(\langle\text { Elementary，my，dear，} \text { Watson }\rangle)}{\text { sup }_{A W_{k}}^{\text {Noise }}(\langle\text { Elementary，} \text { my，dear，} \text { Watson }\rangle)}=\frac{1}{3} \simeq 0.33$ ． Thus，the mining of closed sequential patterns 1－ contiguous $\delta$－robust when $\delta \in[0 ; 0.74]$ allows to remove the pattern 〈Elementary，my，Watson〉 of the resulting set and thus contributes to the improvement of the apprehensibility of the set of patterns．For $\delta \in[0.34 ; 0.74]$ ，the set of extracted patterns becomes： $\langle E l e m e n t a r y, m y: 5,\langle E l e m e n t a r y$, my，dear，Watson $\rangle$ ： 3，〈Watson〉： 5 ．
To summarize，regular sequential pattern mining of－ ten produces an unmanageable set and regularly faces issues related to both execution time and memory us－ age．Contiguous sequential pattern mining allows to im－ prove both the apprehensibility of the resulting set and the efficiency of the mining．However，this mining loses information when the data is noisy．The $k$－contiguous $\delta$－robust sequential pattern mining is a solution that pre－ vents the mining of unexpected additional items in noisy data and that guarantees，to a certain extent，the conti－ guity of patterns．

However，the other two disturbances that are item loss and substitution，remain an issue when mining closed contiguous sequential patterns in noisy data．Indeed， in our running example，with $\delta \in[0.34 ; 0.74]$ ，the re－ sulting set is still made up of 3 patterns（see Table 2）， whereas we only look for the information of the pat－ tern 〈Elementary，my，dear，Watson〉．Therefore，this set contains redundant information，which leads to a loss of apprehensibility for the user．The closedness constraint allows to remove the patterns having a super－ pattern of the same support．

## 4．3．Extended－Closedness Constraint

The problem raised by the loss and substitution of 740 items in data is also the decrease of the support of the ${ }^{741}$ associated patterns．This decrease leads to the pres－${ }^{742}$ ence of sub－patterns in the resulting set that cannot be removed by the closedness constraint．Indeed，in the set of $\mathrm{ClCoS} P$ from the Noise database，the pattern〈Elementary，my，dear，Watson〉 has only a support of 2， while the sub－patterns 〈Elementary，my and 〈Watson〉 are also $\operatorname{ClCoS} P$ with a support of 5 ．Therefore，the closedness constraint cannot remove these sub－patterns． The resulting set thus becomes less apprehensible．As ${ }^{74}$ the loss and substitution of items is irreversible when mining noisy data，our goal is to mitigate the increase of the resulting set of patterns by removing some of the sub－patterns to improve the apprehensibility，at the price of the information regarding the support of these new sub－patterns．To achieve this goal，we propose to re－ lax the closedness constraint to form a new constraint． The resulting extended－closedness constraint does not require that a sub－pattern has the same support than it super－pattern to be removed．The extended ratio $\lambda$ ，as－ sociated to this constraint，allows to set to what extent ${ }^{757}$ the support of a super－pattern can be lower than the sup－${ }^{758}$ port of its sub－patterns for them to be removed．We rely 759 on the definition 3.3 to introduce the following defini－ 760 tion：

Definition 4.4 （Extended－closed sequential pattern）． Given $\lambda \in[0 ; 1]$ a ratio，$S D B$ a sequence database，and $P$ a closed sequential pattern．$P$ is a $\lambda$－closed sequential pattern if and only if there exists no super－pattern $P^{\prime}$ verifying：$\frac{\sup _{A}^{S D B}\left(P^{\prime}\right)}{\sup _{A}^{S D B}(P)} \geq \lambda$ ．

If $P^{\prime}$ exists and is a $\lambda$－closed sequential pattern the reference closed sequential pattern of $P^{\prime}$ is the shortest closed sequential pattern $P_{c} \subset P^{\prime}$ verifying： $\frac{s u p_{A}^{S D B}\left(P^{\prime}\right)}{s u p_{A}^{S D B}\left(P_{c}\right)} \geq \lambda$ ．

We now present in Table 3 the use of the extended－ closedness constraint depending on the value of the extended ratio．This new constraint will contribute to the reduction of the number of mined patterns．The configuration where $\lambda=1$ is similar to the closed contiguous sequential pattern mining like in Table 1．In the configuration where $\lambda=0.6$ ，the pattern〈Elementary，my，dear〉 disappears from the set of ${ }^{779}$ extracted patterns，as：〈Elementary，my，dear〉 $\sqsubset_{c}{ }^{780}$〈Elementary，my，dear，Watson〉 and $\frac{\text { sup }_{A W_{k}}^{\text {Noise }}(\langle\text { Elementary，my，dear，Watson }\rangle)}{\sup _{A W_{k}}^{\text {Noise }}\langle\text { Elementary，my，dear }\rangle} \simeq 0.66$

## 4．4．Notations of the Patterns of Interest

To clarify the set of sequential patterns we are interested in，Table 4 presents the complete list，and the associated abbreviations．

## 4．5．Summary on the Running Example

We now exhaustively list the various sets of se－ quential patterns that can be mined from the sequence database Noise（Table 1）with min＿sup $=2, k=1$ wild－ card，$\delta=0.6$ and $\lambda=0.6$ ．The complete list of patterns is presented in Table 5.
The set of frequent contiguous sequential patterns $\operatorname{CoS} P$ is made up of 10 patterns and decreases to 4 pat－ terns in the set of closed contiguous sequential patterns ClCoS P．Both sets（grayed out in Table 5）are mined using the contiguity and closedness constraints of the literature．

The use of $k=1$ wildcard（ $\mathrm{ClCo}_{1}$ mining）results in a set of the same size（4 patterns）．However，as seen pre－ viously，the 1 －support of several patterns is increased：
－〈Elementary，my，Watson〉，increases from 1 to 4. It thus becomes not only frequent，but also closed．
－〈Elementary，my，dear，Watson〉，increases from 2 to 3 ．

In addition，〈Elementary，my，dear〉 and〈Elementary，my，dear，Watson〉 now have the same support，so 〈Elementary，my，dear〉 is no more closed．
Since $0.66 \geq \lambda=0.6$ ，we consider that both patterns are considered to contain the same information，so〈Elementary，my，dear〉 is not a 0．6－closed contiguous sequential pattern．In the configuration where $\lambda=0.4$ ：
$\frac{\sup _{A W_{k}}^{N o i s e}(\langle\text { Elementary，my，dear，} \text { Watson }\rangle)}{\sup _{A W_{k}}^{N o s_{e} e}(\langle\text { Elementary，my }\rangle)}=$
$\frac{\sup _{A W_{k}}^{\text {Noise }}(\langle\text { Elementary，my，dear，} \text { Watson }\rangle)}{\sup _{A W_{k}}^{\text {Noise }}(\langle\text { Watson }\rangle)}=0.4$ ．
Since $0.4 \geq \lambda=0.4$ ，the only 0.4 －closed contiguous sequential pattern is 〈Elementary，my，dear，Watson〉． A configuration with an extended ratio as low as $\lambda=0.4$ allows a potential large loss of the support information as a 0.4 －closed sequential pattern only needs to have $40 \%$ of the support of its sub－patterns to remove them．
As a $\lambda$－closed sequential pattern is defined as a par－ ticular closed sequential pattern，the complete set of $\lambda$－ closed sequential patterns is a subset of the set of closed sequential patterns．Therefore，the complete set of $\lambda$－ closed $k$－contiguous sequential patterns can be mined while mining closed $k$－contiguous sequential patterns．

Table 3：Impact of the extended－closedness constraint in the mining in Noise database

| $\lambda$ | Cl $\lambda_{\lambda}$ CoS $P$ |
| :--- | :--- |
| 1 | $\langle$ Elementary，my $\rangle: 5,\langle$ Elementary，my，dear $\rangle: 3,\langle$ Elementary，my，dear，Watson $\rangle: 2,\langle$ Watson $\rangle: 5$ |
| 0.6 | $\langle$ Elementary，my $\rangle: 5,\langle$ Elementary，my，dear，Watson $\rangle: 2,\langle$ Watson $\rangle: 5$ |
| 0.4 | $\langle$ Elementary，my ，dear，Watson $\rangle: 2$ |

〈Elementary，my，Watson〉 is mined three times out 824 of four thanks to the use of a wildcard，$\frac{3}{4}=0,75>_{825}$ $\delta=0.6$ ．Thus，this pattern does not respect the $0.6-{ }^{826}$ robustness constraint．The set of $\mathrm{ClCo}_{1} R_{0.6}$ patterns is ${ }_{827}$ thus reduced to 3 patterns．

```
    〈Watson〉 \(\sqsubset_{1}\) 〈Elementary, my, dear, Watson〉 and
\(\frac{\text { sup }_{A W_{k}}^{\text {Noise }_{k}}(\langle\text { Elementary, my, dear, Watson }\rangle)}{\text { sup }^{\text {Noise }}(\langle\text { Watson }\rangle)}=0.6\).
        sup \(_{A W_{k}}^{\text {Noise }}(\langle\) Watson \(\rangle)\)
Likewise,
\(\left\langle\right.\) Elementary, my〉 \(\sqsubset_{1}\langle\) Elementary, my, dear, Watson〉
and \(\frac{\text { sup }_{A W_{k}}^{N o i s e}(\langle\text { Elementary, my, dear, Watson }\rangle)}{\sup _{A W_{k}}^{N o i s e}(\langle\text { Elementary, my }\rangle)}=0.6\).
```

The 0.6-closedness allows to remove both patterns
$\langle$ Watson $\rangle$ and $\langle$ Elementary, my $\rangle$.

The final set of patterns $C l_{0.6} C o_{1} R_{0.6}$ is reduced to one single pattern 〈Elementary，my，dear，Watson〉 with a 1 －support of 3 ，which greatly increases the apprehensibility of the set of patterns，in comparison to the set of $C l C o S P$ ．
In the 5 sequences from the Noise database， $2^{841}$ sequences contain a contiguous occurrence of ${ }^{842}$〈Elementary，my，dear，Watson〉 without noise and 3 of them contain it with disturbances of items of type loss，substitution or addition（actually on of each）．We can conclude that the use of the robustness constraint makes it possible to overcome the addition of items by finding one additional occurrence of〈Elementary，my，dear，Watson〉 and by removing the semi－contiguous sequential patterns appeared thanks to the use of a wildcard．We can say that the use of the extended－closedness constraint makes it possible to disregard the patterns appearing due to the loss and the substitution of items in the Noise database．

In conclusion，both new robustness and extended－ 856 closedness constraints improve the apprehensibility of ${ }_{857}$ the set of $\mathrm{ClCoS} P$ mined from noisy data．Indeed，on ${ }_{858}$ the one hand，they mitigate the addition of items in data，${ }_{859}$ while preserving the contiguity constraint as much as 860 possible．On the other hand，they increase the apprehen－${ }^{861}$ sibility of the resulting set，by avoiding segmentations in 862 patterns．

## 4．6．Problem statement

As part of our problem，which is to improve the ef－ ficiency of the pattern mining and the apprehensibility of its results，we want to propose a sequential pattern mining algorithm that natively integrates $\lambda$－closedness， $k$－contiguity，$\delta$－robustness constraints，and allows for an easy integration of a set of monotonic or anti－monotonic constraints．Finally，the algorithm must answer the following problem：given $S D B$ a sequence database， min＿sup a minimum support threshold，$k$ a number of wildcards，$\delta$ a robustness ratio，$\lambda$ an extended ratio and $\zeta$ a set of monotonic or anti－monotonic constraints，how to mine $\mathrm{Cl}_{\lambda} \mathrm{Co}_{k} R_{\delta} C_{\zeta} S P$ ，the complete set of $\lambda$－closed $k$－ contiguous $\delta$－robust sequential patterns satisfying a set of constraints $\zeta$ ？

## 5．$C l_{\lambda} C o_{k} R_{\delta} C_{\zeta} S P$ Mining

In this section，we introduce the C3Ro algorithm dedicated to $\mathrm{Cl}_{\lambda} \mathrm{Co}_{k} R_{\delta} C_{\zeta} S P$ mining．The name C3Ro stands for Closed Contiguous Robust Constrained se－ quential pattern mining algorithm．C3Ro is based on the pattern－growth philosophy，used in many sequen－ tial pattern mining algorithms，such as PrefixSpan［27］， BIDE［47］，PG［44］，CCPM［2］，etc．C3Ro natively in－ tegrates closedness（as Clospan［51］）and contiguity（as CCSpan［58］）constraints，in addition to any monotonic or anti－monotonic constraints（as PG［44］）．All these constraints can be easily managed thanks to the pattern－ growth philosophy．
The main idea behind C3Ro is to provide a toolbox al－ lowing the final user to mine multi－constraints sequen－ tial patterns that fit his／her needs，through the use of sev－ eral meaningful parameters：
－$k$ a maximum number of wildcards per pattern oc－ currence．
When $k=0$ ，C3Ro mines contiguous sequential patterns and when $k=\infty$ ，it mines regular se－ quential patterns．More importantly，the number of wildcards is a way to mine semi－contiguous se－ quential patterns or to improve the mining of con－ tiguous sequential patterns in noisy data．

- $\delta$ a robustness ratio, to mine the newly introduced $k$-contiguous $\delta$-robust sequential patterns. It allows C3Ro to limit the impact of wildcards on the set of occurrences of each pattern, thus on the resulting set of patterns, by adjusting the robustness ratio on the estimated amount of noise in the data.
- $\lambda$ an extended ratio, to mine the newly introduced 904 $\lambda$-closed sequential patterns.
It allows C3Ro to alleviate the closedness constraint with the extended ratio, and thus to extract a more compact set of patterns than the traditional set of closed sequential patterns. When $\lambda=0$ C3Ro mines maximal sequential patterns and when ${ }^{9}$ $\lambda=1, \mathrm{C} 3$ Ro mines closed sequential patterns.
- $\zeta$ a set of monotonic or anti-monotonic constraints.

In Table 6, we can see the adjustment of the extended ratio $\lambda$ and the maximum number of wildcards $k$ allows the C3Ro algorithm to extract numerous types of pattern. We do not make the robustness ratio vary $(\delta=1)$ as it does not impact the type of the patterns mined.

C3Ro is able to mine a wide variety of patterns, depending on the number wildcards, the robustness ratio, the extended ratio and a set of monotonic or anti-monotonic constraints integrated in the mining. All theses parameters have a non negative impact on the efficiency of C3Ro (time and memory), therefore the efficiency of C3Ro is at worst the one of BIDE.

We introduce the sequence database $S D B$ (Table 7) to illustrate, throughout this section, how C3Ro algorithm works. The structure of C3Ro is the following:
First, C3Ro uses a framework that enumerates frequent $k$-contiguous $\delta$-robust sequential patterns satisfying a set of monotonic or anti-monotonic constraints. This framework is guided by the one of PrefixSpan to enumerate frequent sequential patterns. Second, C3Ro uses a new extended-closedness checking scheme, a pruning technique and a Backscan

Table 7: Sequence database $S D B$

| sid | sequence |
| :--- | :--- |
| \#1 | $\langle\{a\}\{b\}\{c\}\{b d\}\rangle$ |
| \#2 | $\langle\{a\}\{b\}\{c\}\rangle$ |
| \#3 | $\langle\{a b\}\{d\}\{b\}\{c\}\{d\}\rangle$ |

method, guided by the BIDE algorithm. We now present both elements.

### 5.1. Framework to Enumerate Frequent $k$-Contiguous $\delta$-Robust Sequential Patterns.

We start by introducing some definitions related to the projected database with $k$ wildcards (in line with the definition of a contiguous projected database) and the prefix-monotone constraint, required by the framework we propose. Then we explain how to enumerate frequent $k$-contiguous $\delta$-robust sequential patterns and illustrate it on our running example (Table 7).

### 5.1.1. Preliminaries

Given $S D B$ an input database, $S=\left\langle E_{1}^{\prime}, E_{2}^{\prime}, \ldots, E_{m}^{\prime}\right\rangle$ an input sequence and $P=\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$ a sequential pattern with $P \sqsubseteq_{k} S$. By definition, there exist integers $j_{1}, j_{2}, \ldots, j_{n}$ such that: (1) $1 \leq j_{1}<j_{2}<\ldots<j_{n}<m$ ; and (2) $E_{1} \subseteq E_{j_{1}}^{\prime}, E_{2} \subseteq E_{j_{2}}^{\prime}, \ldots, E_{n} \subseteq E_{j_{n}}^{\prime}$; and (3) $j_{n}-j_{1}-n \leq k$ (see definition 4.3):

Definition 5.1 (Number of wildcards available). To extend the $k$-prefix $P$ in the sequence $S$, at least one wildcard has to be available. This number is equal to $k-\left(j_{n}-j_{1}-n\right)$.

Definition 5.2 (Projected $k$-items of a sequential pattern). The projected $k$-items of $P$ in $S$ are the union of two sets 1) the set of i-extensions $e_{i}$ of each occurrence of $P$ in $S$ such that $E_{n} \diamond_{i} e_{i} \subseteq E_{j_{n}}^{\prime} 2$ ) the set of s-extensions $e_{s}$ of each occurrence of $P$ in $S$ such that $e_{s} \in E_{j_{n+1}}^{\prime} \cup E_{j_{n+2}}^{\prime} \ldots \cup E_{\min \left(j_{n+\left(k-j_{n}-j_{1}-n\right)+1}, m\right)}^{\prime}$.

Definition 5.3 (Projected $k$-sequence). The set of projected $k$-items of $P$ in $S$ is called the projected $k$ sequence of $P$ in $S$.

Definition 5.4 ( $k$-prefix sequential pattern). $P$ is called the $k$-prefix sequential pattern.

Definition 5.5 (Projected $k$-database). The set of projected $k$-sequences of $P$ in $S D B$ is called the projected $k$-database of $P$ in $S D B$, denoted $P_{-k} S D B$.

Definition 5.6 (Usage of wildcards). For each $j_{i}$ with 990 $i \in \llbracket 1 ; n\left[\right.$, if $j_{i+1}-j_{i}>1$, we write $P={ }_{991}$ $\left\langle E_{1}, E_{i}, E_{i+1}{ }^{*}, \ldots, E_{n}\right\rangle$ to identify itemsets found with the 992 use of a wildcard in a projected $k$-database.

In our running example, in the sequence $(a)(b)(c)$ the set of contiguous projected items (with no wildcards) of the prefix sequential pattern $(a)$ is: $\left\{\{(b)\},\{(b)\},\left\{\left(\_b\right),(d)\right\}\right\}$.

If a wildcard is used to find an item, a $*$ is denoted on the item it corresponds to. In our example, when $k=1$, and no wildcard is used yet, all the first items of 999 second adjacent itemset after (a) are projected items too: 1000 $\left\{\left\{(c)^{*}\right\},\left\{(c)^{*}\right\},\left\{(b)^{*}\right\}\right\}$. To summarize, the set of projected 1001 1-items of the prefix sequential pattern $(a)$ in $(a)(b)(c){ }^{1002}$ is $\left\{\left\{(b),(c)^{*}\right\},\left\{(b),(c)^{*}\right\},\left\{(-b),(d),(b)^{*}\right\}\right\}$.
Definition 5.7 (Wildcard support in a projected $k$-database). Given $P$ a $k$-prefix sequential pattern and $e$ an item of its projected $k$-database. The wildcard support of $e$ is the number of sequences in which $e$ occurs, using at least one wildcard and following an occurrence of $P$ that has used no wildcard.

We illustrate the wildcard support in a projected $k$-database on our running example, with $k=2$ and the pattern $(a)(b)(d)$ :
In sequence sid \#1, the occurrence of $(a)(b)$ has not used any wildcard. One wildcard has to be used to find $(d)$ and form $(a)(b)(d)$. The used wildcard increases the 2 -support by 1 . In sequence id \#3, the 2 -prefix sequential pattern $(a)(b)$ has already used one wildcard and needs to use a second wildcard to find (d). Thus, this wildcard does not increase the wildcard support of $(d)$ in the projected 2-database of $(a)(b)$. Therefore, $\sup _{W_{2}}^{(a)(b) S D B}((d))=1$

The following definition and lemma come from [44]. 1021
Definition 5.8 (Prefix-monotone constraint). A constraint $C$ is called prefix anti-monotonic if for each sequential pattern $P$ satisfying the constraint, every prefix of $P$ also satisfies $C$. A constraint $C$ is called prefix- ${ }^{1024}$ monotonic, if for each sequential pattern $P$ satisfying ${ }^{1025}$ the constraint $C$, each sequence having $P$ as a prefix, ${ }^{1026}$ also satisfies $C$. A constraint is called prefix-monotone ${ }^{1027}$ if it is either prefix anti-monotonic or prefix monotonic. ${ }^{1028}$

## Lemma 5.1 (Prefix-monotone constraint principle). ${ }^{1030}$

 Given a prefix-monotone constraint $C$ and a sequential ${ }_{1031}$ pattern $P$.1. When $C$ is prefix anti-monotonic, if $C(P)=$ false, 1033 then there exists no sequential patterns $P^{\prime}$ that has $P$ as ${ }_{1034}$ a prefix with: $C\left(P^{\prime}\right)=$ true.
2. When $C$ is prefix monotonic, if $C(P)=$ true, then every sequential pattern $P^{\prime}$ having $P$ as a prefix verifies: $C\left(P^{\prime}\right)=$ true .

Theorem 5.2 (Pruning in a projected $k$-database). Given a k-prefix sequential pattern $P$ in a sequence database $S D B$ and $e$ an item of the projected $k$-database of $P$. If an item $e^{\prime}$ always appears after $e$ in the projected $k$-database of $P, e^{\prime}$ can be safely removed from the projected $k$-database.

Proof. Given $e$ and $e^{\prime}$ two items, $P$ a $k$-prefix sequential pattern and $S D B$ a sequence database. If each occurrence of an itemset $e^{\prime}$ always appears after an occurrence of an itemset $e$ in each projected $k$-sequence of $P$, it means that there exists a sequential pattern $Q=\left\langle P \diamond e \diamond e^{\prime}\right\rangle$ with $\sup _{U W_{k}}^{S D B}(Q)=\sup _{U W_{k}}^{S D B}\left(\left\langle P \diamond e^{\prime}\right\rangle\right)$. Thus, each occurrence of the sequential pattern $\left\langle P \diamond e^{\prime}\right\rangle$ is included in an occurrence of $Q$ and so are its extensions. Therefore, it is useless to extend $P$ with $e^{\prime}$, only with $e$ and $e^{\prime}$ can be pruned from the projected $k$-database of $P$.

Theorem 5.3 ( $k$-contiguous $\delta$-robust sequential pattern projection). Given $\delta \in[0 ; 1]$ the robustness ratio, $P$ a $k$-contiguous $\delta$-robust sequential pattern and $e$ a frequent item from its projected $k$-database. $\langle P \diamond$ $e\rangle$ is a $k$-contiguous $\delta$-robust sequential pattern if $\frac{\sup _{W_{k}}^{S D B}(P)+\sup _{W_{k}}^{P S D B}(e)}{\sup _{A W_{k}}^{P S D B}(e)} \leq \delta$.

Proof. Given $P$ a $k$-contiguous $\delta$-robust sequential pattern and $e$ a frequent item from its projected database with $\sup _{W_{k}}^{S D B}(P)+\sup _{W_{k}}^{P-k S B}(e) \leq \sup _{A W_{k}}^{P-k S B}(e) *$ $\delta$. Building on the definition 5.7, $\sup _{W_{k}}^{S D B}(P)+$ $\sup _{W_{k}}^{P k S D B}(e)=\sup _{W_{k}}^{S D B}(\langle P \diamond e\rangle)$. As $\sup _{A W_{k}}^{P k S D B}(e)=$ $\sup _{A W_{k}}^{S D B}(\langle P \diamond e\rangle)$, if $\sup _{W_{k}}^{S D B}(\langle P \diamond e\rangle) \leq \sup _{A W_{k}}(e) * \delta$ then $\langle P \diamond e\rangle$ is a $k$-contiguous $\delta$-robust sequential pattern.

### 5.1.2. Enumeration of Frequent $\mathrm{Co}_{k} R_{\delta} C_{\zeta} S P$.

In C3Ro, the complete search space of $k$-contiguous sequential patterns forms a sequential pattern tree, as in [7]. The root node of the tree is at the top level and is labeled $\emptyset$. To form the tree, C3Ro recursively extends a node (referred to as a $k$-prefix sequential pattern) at a certain level in the tree by adding at most $k+1$ contiguous itemsets per occurrence of the $k$-prefix sequential pattern.
As C3Ro is designed to mine patterns satisfying $\zeta$ a set of monotonic and/or anti-monotonic constraints, the framework only extends the $k$-prefix sequential pattern with frequent items according to Lemma 5.1, depending
on the given constraint(s) in the $\zeta$ set. Applying this ${ }_{1085}$ Lemma greatly reduces the search space, hence the ${ }_{1086}$ efficiency of C3Ro. It also reduces the resulting set, 1087 by removing irrelevant patterns. The Lemma is used ${ }_{1088}$ in C3Ro in the following way: when all frequent ${ }_{1089}$ items satisfying $\zeta$ are identified, each one becomes ${ }_{1090}$ a $k$-prefix sequential pattern to be extended (at this ${ }_{1091}$ step the algorithm is at the first level of the tree). The ${ }_{1092}$ framework then builds the projected $k$-database of each 1093 $k$-prefix. The wildcard support of all the items must be evaluated in order to apply Theorem 5.3. Finally, only ${ }_{10}$ the frequent items in the projected $k$-database with a ${ }_{1096}$ wildcard support verifying Theorem 5.3 can extend the $k$-prefix in $k$-contiguous $\delta$-robust sequential patterns. This process is recursively repeated for each $k$-prefix to mine the resulting set of $C o_{k} R_{\delta} C_{\zeta} S P$.

Now we rely on our running example to describe how to enumerate frequent closed $k$-contiguous sequential patterns with $k=1$ wildcard. The 1-prefix sequential pattern $P=(a)$ is the first one studied (due to the lexicographic order). Its projected 1-database is: $\left\{\left\{(b),(c)^{*}\right\},\left\{(b),(c)^{*}\right\},\left\{(-b),(d),(b)^{*}\right\}\right\}$. In this pro- ${ }_{1106}$ jected 1-database, (b) has an 1-support of 3 and (c) ${ }_{1107}$ has an 1 -support of 2 . As both 1 -support exceed ${ }_{108}$ min_sup, they can extend (a). However, in the pro- ${ }_{1109}$ jected 1-database, $(b)$ always occurs before $(c)^{*}$. According to Theorem 5.8, (a) can be extended with (b), 1110 while (c) can be removed. Then, given the 1-prefix ${ }_{1111}$ sequential pattern $(a)(b)$, its projected 1-database is: ${ }^{1112}$ $\left\{\left\{(c),(b)^{*}\right\},\{(c)\}\{(c)\}\right\} .(c)$ has a 1-support of 3 and can ${ }^{1113}$ extend $(a)(b)$. The projected 1-database of $(a)(b)(c)$ is: 1114 $\{\{((b),(d)\}),\{\emptyset\}\{(d)\}\} .(d)$ is the last item that can ex- ${ }_{1115}$ tend $(a)(b)(c)$. The 1-prefix sequential pattern $(a)$ is ${ }_{1116}$ now fully extended. The same method can be used with ${ }_{1117}$ $P=(b), P=(c)$ and $P=(d)$.

### 5.2. BI-Directional $\lambda$-closedness Checking with Wild- ${ }^{1120}$

 cards.Before explaining the two main steps that are forward-extensions and backward-extensions, we start by explaining global strategy adopted by C3Ro for checking closedness and $\lambda$-closedness. Then, we detail the Backscan pruning method.

### 5.2.1. Closedness and $\lambda$-closedness checking strategy

The previous framework mines $k$-contiguous $\delta$-robust frequent sequential patterns, satisfying prefix-monotone ${ }_{1130}$ constraints. To get the set of frequent $\lambda$-closed se- ${ }_{1131}$ quential patterns, an $\lambda$-closedness checking has to be ${ }_{1132}$ performed. C3Ro $\lambda$-closedness checking scheme is ${ }_{1133}$
based on the same efficient BI-Directional design than BIDE to check closed sequential patterns.

As the definition 4.4 of $C l_{\lambda} C o_{k} S P$ is based on the definition 3.3 of $C l S P$, we first must identify them to be able to identify $C l_{\lambda} C o_{k} S P$. We introduce and redefine here several concepts and theorems of the BIDE [47] algorithm in order to use them in the context of $\mathrm{Cl}_{\lambda} \mathrm{Co}_{k} R_{\delta} C_{\zeta} S P$ mining:
According to the definition 4.4, if $P=\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$ a $C_{o} S P$, is not closed, it means that there exists at least one item $e$, which can be used to extend $P$ to get $P^{\prime}$ a $C o k S P$, with $\sup _{A W_{k}}(P)=\sup _{A W_{k}}\left(P^{\prime}\right)$. In line with BIDE, $P$ can be extended into $P^{\prime}$ in three ways: (1) $P^{\prime}=\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle \diamond e(2) P^{\prime}=e \diamond\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$ and (3) $\exists i, 1<i<n, P^{\prime}=\left\langle E_{1}, E_{2}, \ldots, E_{i}\right\rangle \diamond e \diamond_{s}\left\langle E_{i+1}, \ldots, E_{n}\right\rangle$. (1) $e$ occurs after the itemset $E_{n}$, BIDE challenges $e$ a forward-extension item of $P$. (2) and (3) $e$ occurs before itemset $E_{n}$, BIDE calls $e$ a backward-extension item of $P$.

Theorem 5.4 (BI-Directional extended-closedness checking). $P$ a sequential pattern is closed, if and only if $P$ has no backward-extension and no forwardextension items.

The closedness checking strategy BIDE relies on forward-extension and backward-extension detection. We use the same strategy in the context of $C l_{\lambda} C o_{k} R_{\delta} C_{\zeta} S P$.

Similarly to $C l S P$, if a closed sequential pattern $P=$ $\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$, is not an $C l_{\lambda} C o_{k} S P$, it means that there exists at least one item $e$, which can be used to extend $P$ to get a new sequential pattern $P^{\prime}$, with $\sup _{A W_{k}}\left(P_{c}\right) * \lambda \leq$ $\sup _{A W_{k}}\left(P^{\prime}\right)$ (where $P_{c}$ is the reference closed sequential pattern of $P^{\prime}$ with $P_{c} \sqsubseteq_{k} P$ ). The closed sequential pattern $P$ can be extended into $P^{\prime}$ in three ways: (1) $P^{\prime}=\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle \diamond e(2) P^{\prime}=e \diamond\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$ and (3) $\exists i, 1<i<n, P^{\prime}=\left\langle E_{1}, E_{2}, \ldots, E_{i}\right\rangle \diamond e \diamond_{s}\left\langle E_{i+1}, \ldots, E_{n}\right\rangle$. (1) $e$ occurs after the itemset $E_{n}$, we call $e$ a $\lambda$-forwardextension item of $P$. (2) and (3) $e$ occurs before itemset $E_{n}$, we call $e$ a $\lambda$-backward-extension item of $P$. Given the above elements, the following theorem becomes obvious, according to the definition of a frequent $\lambda$-closed sequential pattern with wildcards (definition 4.4).

Theorem 5.5 (BI-Directional extended-closedness checking). $P$ a $C o_{k} S P$ is a $C l C o o_{k} S P$, if and only if $P$ has no $\lambda$-backward-extension and no $\lambda$-forwardextension items.

We have just shown that thanks to the four types of ${ }_{1175}$ extension we can check if a pattern is closed and then if ${ }_{1176}$ it is $\lambda$-closed.
We propose to follow the procedure below to identify the $\lambda$-closed sequential patterns when enumerating the $\mathrm{Co}_{k} R_{\delta} C_{\zeta} S P$. We evaluate each $k$-prefix $C o_{k} R_{\delta} C_{\zeta} S P_{1180}$ that has just been validated by the enumeration framework to first determine if this $k$-prefix is closed. Then ${ }_{11}$ the $k$-prefix is kept if lambda-closed, if not, a superpattern necessary is. Thus, the support of the reference closed sequential pattern is kept to find the $\lambda$-closed sequential pattern in a future extension of this $k$-prefix.

We now introduce the lemmas to identify the four types of extension and evaluate the closedness and the ${ }^{1190}$ $\lambda$-closedness of a pattern. We start with both lemmas to ${ }^{1191}$ check forward-extension and $\lambda$-forward-extension:

Lemma 5.6 (Forward-extension item checking). Given ${ }^{1194}$ $P$ a ClCo $_{k} S$ P and SDB a sequence database, its com- ${ }^{1195}$ plete set of forward-extensions items is the set of items e ${ }^{1196}$ in its projected $k$-database that verifies: $\sup _{A W_{k}}^{P_{-k} S D B}(e)={ }^{1197}$ $\sup _{A W_{k}}^{S D B}(P)$.

Proof. For each item $e$ in the projected $k$-database of $P$, a new $C o o_{k} S P P^{\prime}=\langle P \diamond e\rangle$ can be cre- ${ }^{12}$ ated, if $\sup _{A W_{k}}^{P_{-k} S D B}(e)=\sup _{A W_{k}}^{S D B}(P)$ thus $\sup _{A W_{k}}^{S D B}\left(P^{\prime}\right)={ }^{1201}$ $\sup _{A W_{k}}^{S D B}(P)$ then $P$ is not a $C l C o{ }_{k} S P$ and $\langle e\rangle$ is a forwardextension of $P$.

Lemma 5.7 ( $\lambda$-forward-extension item checking). ${ }^{1206}$ Given PaClCo ${ }_{k} S P, P_{c}$ its reference closed sequential ${ }_{1207}$ pattern and S DB a sequence database, its complete set ${ }^{1208}$ of $\lambda$-forward-extensions items is the set of items e in its ${ }^{1209}$ projected $k$-database that verifies: $\frac{\sup _{A W_{k}}^{P_{-} S D B}(e)}{\sup _{A W_{k}}^{S D B}\left(P_{c}\right)}>\lambda$.

Proof. For each item $e$ in the projected $k$-database of $P$, a new $C o_{k} S P P^{\prime}=\langle P \diamond e\rangle$ can be created, if $\frac{\sup _{A W_{k}}^{P_{-} S D B}(e)}{\sup _{A W_{k}}^{S D B}\left(P_{c}\right)}>\lambda$ thus $\frac{\sup _{A W_{k}}^{P S D B}\left(P^{\prime}\right)}{\sup _{A W_{k}}^{S D B}\left(P_{c}\right)}>\lambda$ then $P$ is not a $C l_{\lambda} C o_{k} S P$ and $\langle e\rangle$ is a $\lambda$-forward-extension of $P$.

These lemmas show that the evaluation of the $k$ support of items in the projected $k$-database of a $k$-prefix allows to determine if this $k$-prefix has both a forward- ${ }_{1220}$ extension and a $\lambda$-forward-extension.

### 5.2.3. Backward-extensions and $\lambda$-backwardextensions checking

We now present how to identify backward-extensions and $\lambda$-backward-extensions. These extensions combine two ways to extend the $k$-prefix: either the item appears between the first and last itemset of the $k$-prefix, or the item appears before the first itemset of the $k$-prefix. We introduce a theorem allowing a pruning of extensions between the first and the last itemset of the $k$-prefix, during the enumeration phase.

Theorem 5.8. Given $P$ a $C o_{k} R_{\delta} C_{\zeta} S P, S D B$ a sequence database and $e_{1}$ an item of the projected $k$-database of $P$. If an item $e_{2}$ always appears after $e_{1}$ in each projected $k$-sequence of $P$ then $e_{2}$ can be removed from the projected $k$-database of $P$.

Proof. Given $P$ a $C o o_{k} R_{\delta} C_{\zeta} S P, S D B$ a sequence database and $e_{1}, e_{2}$ two items of the projected $k$ database of $P$. If $e_{2}$ always appears after $e_{1}$ in each projected $k$-sequence of $P$, then it means that there exists $P^{\prime}$ a $C o_{k} R_{\delta} S P$ such as $P^{\prime}=\left\langle P \diamond e_{1} \diamond e_{2}\right\rangle$ with $\sup _{U W_{k}}^{S D B}\left(P^{\prime}\right)=\sup _{U W_{k}}^{S D B}\left(\left\langle P \diamond e_{2}\right\rangle\right)$. Each occurrence of $\left\langle P \diamond e_{2}\right\rangle$ is included in an occurrence of $P^{\prime}$, thus all its extensions are also included the extensions of $P^{\prime}$. Therefore, it is useless to extend $P$ with $e_{2}$ and $e_{2}$ can be removed from the projected $k$-database of $P$.

Thanks to this theorem, $\lambda$-backward-extensions between the first and last itemset of the $k$-prefix are pruned during the enumeration phase, so we can focus only on identifying backward-extensions and $\lambda$-backwardextensions when the item appears before the first itemset of the $k$-prefix. Given $S D B$ a sequence database, $S$ a sequence $\left\langle E_{1}^{\prime}, E_{2}^{\prime}, \ldots, E_{m}^{\prime}\right\rangle$ from $S D B$ and $P$ a $C o_{k} R_{\delta} C_{\zeta} S P$ $\left\langle E_{1}, E_{2}, \ldots, E_{n}\right\rangle$ in $S D B$ such as $P \sqsubseteq_{k} S$. It means there exists integers $j_{1}, j_{2}, \ldots, j_{n}$ such as: (1) $1 \leq j_{1}<j_{2}<$ $\ldots<j_{n} \leq m ;(2) E_{1} \subseteq E_{j_{1}}^{\prime}, E_{2} \subseteq E_{j_{2}}^{\prime}, \ldots, E_{n} \subseteq E_{j_{n}}^{\prime} ;(3)$ $j_{n}-j_{1}-n \leq k$. The number $k-\left(j_{n}-j_{1}-n\right)$ represents the number of wildcards available for $P$.

Definition 5.9 (Backward $k$-sequence). The backward $k$-sequence of $P$ in $S$ is the union of the set of iextensions $e_{i}$ such as $e_{i} \diamond_{i} E_{1} \subseteq E_{j_{1}}^{\prime}$ and the set of s-extensions $e_{s}$ such as $e_{s} \in E_{j_{1}-1}^{\prime} \cup E_{j_{1}-2}^{\prime} \cdots \cup$ $E_{\max \left(j_{1}-\left(k-\left(j_{n}-j_{1}-n\right)\right)-1,1\right)}^{\prime}$, denoted $P_{-B_{k}} S$.

Definition 5.10 (Backward $k$-database). The set of backward $k$-sequences of $P$ in $S D B$ is called the backward $k$-database of $P$ in $S D B$, denoted $P_{-B_{k}} S D B$.

We introduce the two lemmas to identify the backward-extensions and $\lambda$-backward-extensions:

Lemma 5.9. Given $P$ a $\mathrm{ClCo}_{k} S P$ and $S D B$ a sequence ${ }_{1267}$ database, its complete set of forward-extensions items is the set of items e in its projected $k$-database that verifies: $\sup _{A W_{k}}^{P_{-B_{k}} S D B}(e)=\sup _{A W_{k}}^{S D B}(P)$.
Proof. For each item $e$ in the backward $k$-database ${ }^{1271}$ of $P$, a new $C o_{k} S P P^{\prime}=\langle e \diamond P\rangle$ can be created, ${ }_{1272}$ if $\sup _{A W_{k}}^{P_{-B_{k}} S D B}(e)=\sup _{A W_{k}}^{S D B}(P)$ thus $\sup _{A W_{k}}^{S D B}\left(P^{\prime}\right)={ }^{1273}$ $\sup _{A W_{k}}^{S D B}(P)$ then $P$ is not a $C l C o o_{k} S P$ and $\langle e\rangle$ is a ${ }^{1274}$ backward-extension of $P$.

Lemma 5.10. Given a $\mathrm{ClCo}_{k} S P$ P and its closed se- ${ }_{1277}$ quential pattern $P_{c}$ in a sequence database $S D B$, its ${ }_{1278}$ complete set of $\lambda$-backward-extensions items is the set ${ }_{1279}$ of items $e$ in its backward $k$-database that verifies: ${ }_{1280}$ $\frac{\sup _{A W_{k}}^{P_{-B_{k}} S D B}(e)}{\sup _{A W_{k}}^{S D B}\left(P_{c}\right)}>\lambda$.
Proof. For each item $e$ in the backward $k$-database of ${ }_{1284}$ $P$, a new $C o{ }_{k} S P P^{\prime}=\langle e \diamond P\rangle$ can be created, if $\frac{\sup _{A W_{k}}^{P_{-B_{k}} S D B}(e)}{\sup _{A W_{k}}^{S D B}\left(P_{c}\right)}>\lambda$ thus $\frac{\sup _{A W_{k}}^{P_{-B_{k}} S D B}\left(P^{\prime}\right)}{\sup _{A W_{k}}^{S D B}\left(P_{c}\right)}>\lambda$ then $P$ is
not a $C l_{\lambda} C o_{k} S P$ and $\langle e\rangle$ is a $\lambda$-backward-extension of ${ }_{1287}$ $P$.

The theorem 5.8 and the lemmas 5.9 and 5.10 show ${ }^{1289}$ that the evaluation of the $k$-support of items in the ${ }^{129}$ backward $k$-database of a $k$-prefix allows to identify ${ }^{1291}$ its backward-extensions and $\lambda$-backward-extensions. Therefore, the strategy we propose to verify the closed- ${ }^{12}$ ness and the $\lambda$-closedness of a pattern is based on the ${ }^{1294}$ evaluation of the $k$-support of the items in the backward ${ }^{1295}$ $k$-database of each $k$-prefix.

### 5.2.4. Backscan pruning method

A new Backscan pruning method has been designed ${ }^{1300}$ in order to avoid mining non-closed sequential patterns. ${ }^{1301}$ The idea behind the Backscan pruning is to identify ${ }^{1302}$ items that always appear contiguously with the $k$-prefix ${ }^{1303}$ in each backward sequence.
Theorem 5.11 (Backscan pruning). Given a $k$-prefix $P^{1305}$ in a sequence database $S D B$. If an item $e$ is always the first item before each occurrence of $P$, then $P$ can be safely stopped to be extended.

Proof. Given an item $e$, a $k$-prefix $P$ and a sequence ${ }_{1310}$ database $S D B$. If an item $e$ is always the first item be- 1311 fore each occurrence of $P$, it means that there exists a ${ }_{1312}$ $C o_{k} S P Q=\langle e \diamond P\rangle$ using the same number of wild- ${ }^{1313}$ cards as $P$ with $\sup _{U W_{k}}^{S D B}(Q)=\sup _{U W_{k}}^{S D B}(P)$. Thus, the ${ }_{1314}$ sequential pattern $P$ is always included in $Q$ and so are ${ }_{1315}$ its extensions and it is useless to extend $P$.

### 5.3. Illustration of $\mathrm{Cl}_{\lambda} \mathrm{Co}_{k} R_{\delta} C_{\zeta} S P$ mining with C3Ro

In the previous section, we have introduced the definitions and theorems necessary to enumerate the $C o_{k} R_{\delta} C_{\zeta} S P$ (section 5.1.2) and then to evaluate their $\lambda$-closedness (section 5.1.3). Before presenting Algorithm C3Ro, we first illustrate its process. We rely on $S D B$ sequences (Table 7) to extract the $C l_{0.65} C o_{1} R_{0.6}$ patterns with min_sup $=2$.
Items $a: 3, b: 3, c: 3$ and $d: 2$ are frequent. In alphabetical order, the 1 -prefix $P=\langle\{a\}\rangle$ is the first pattern to be extended. The first step is the enumeration of the $C_{0} R_{0.6} S P$ of 1-prefix $P$. The projected 1database of $P$ is $\left\{\left\langle\{b\},\left\{c^{*}\right\}\right\rangle,\left\langle\{b\},\left\{c^{*}\right\}\right\rangle,\left\langle\{-b\},\{d\},\left\{b^{*}\right\}\right\rangle\right\}$. Items $b$ and $c$ are frequent in the projected 1-database of $\langle\{a\}\rangle$. We notice that item $c$ is always preceded by item $b$ in the projected 1-database of $\langle\{a\}\rangle$. In accordance with Theorem 5.8, item $c$ can therefore be removed from the projected 1-database. We also note that $\frac{s u p_{W_{k}}^{S D B}(\langle\{a\}\rangle)+\sup _{W}^{M_{\star} S D B}(c)}{\sup _{A W_{k}}^{M_{-1} S D}(c)}=\frac{0+2}{2}=1>0.6$, according to Theorem 5.3 item $c$ can also be removed from the projected 1-database of $P=\langle\{a\}\rangle$. Theorem5.8 allows pruning by identifying if one item always precedes another, while Theorem 5.3 evaluates the $k$-support and wildcard $k$-support of each item and prunes items that cannot form $\delta$-robust sequential patterns. The second step is the evaluation of the 0.65 -closedness of the 1 prefix $P$. Knowing that item $b$ has a 1 -support of 3 , just like the 1-prefix $P=\langle\{a\}\rangle$, according to Lemma 5.6, this 1 -prefix is neither closed nor 0.65 -closed.

We now consider the 1-prefix $P=\langle\{a\},\{b\}\rangle$ with the two same steps: enumeration of the $\mathrm{Co}_{1} R_{0.6} S P$ of 1-prefix $P$ and evaluation of the 0.65 -closedness of the 1-prefix $P$. The projected 1-database of $P$ is: $\left\{\left\langle\{c\},\left\{b d^{*}\right\}\right\rangle,\langle\{c\}\rangle,\left\langle\{c\},\left\{d^{*}\right\}\right\rangle\right\}$. Items $\{c\},\{d\}$ are frequent. Item $d$ is always preceded by item $c$ in the projected 1-database, so it is useless to keep $d$ in the projected 1-database. Item $c$ has a 1 -support of 3 , just like $P$, the pattern $P=\langle\{a\},\{b\}\rangle$ is neither closed nor 0.65closed.

We consider the 1-prefix $P=\langle\{a\},\{b\},\{c\}\rangle$ and repeat the same two steps, its projected 1-database is: $\{\langle\{b d\}\rangle,\langle\{d\}\rangle\}$. Item $d$ is frequent, so it can extend the prefix. There are no items of the same 1 -support as the 1 -prefix in the projected 1-database and therefore no forward-extensions. Since the backward 1database of the 1 -prefix is empty because it has no item before item $a$, therefore $P$ has no backwardextensions either. According to Theorem 5.4 , the $\langle\{a\},\{b\},\{c\}\rangle$ pattern is closed. According to Lemma $5.7, d$ is a 0.65 -forward-extension of $\langle\{a\},\{b\},\{c\}\rangle$ be-
cause the 1 -support of $d$ in the projected 1-database checks: $\frac{\sup _{A W_{k}}^{\langle\{a\},\{b\rangle,\{c\}\rangle-B_{1} S D B}(d)}{\sup _{A W_{k}}^{S D B}(\langle\{a\},\{b\},\{c\}\rangle)}=\frac{2}{3}=0.66 \geq \lambda=0.65$ Therefore, $\langle\{a\},\{b\},\{c\}\rangle$ is not 0.65 -closed.

The 1-prefix $P=\langle\{a\},\{b\},\{c\},\{d\}\rangle$ is now considered. This 1-prefix has neither a backward 1-database nor a projected 1-database, because no item occur before or after $P$. Thus, $P$ has no extension. According to Theorem 5.5, $\langle\{a\},\{b\},\{b\},\{c\},\{d\}\rangle$ is therefore the only 0.65 -closed sequential pattern having as first item $a$.

The same method can be applied with the 1-prefixes $P=\langle\{b\}\rangle, P=\langle\{c\}\rangle$ and $P=\langle\{d\}\rangle$. Finally, $C l_{0.65} C o s_{1} R_{0.6} S P$ is made up of only one pattern: $\{\langle\{a\},\{b\},\{c\},\{d\}\rangle\}$ compare to the $9 \operatorname{CoS} P$.

To summarize, in the frame of a $\mathrm{ClCoS} P$ mining, the use of $k=1$ wildcard allows to discover new patterns that may not appear in the resulting set due to noisy data ( 2 patterns). The 0.6 -robustness constraint sets the limit between contiguous sequential patterns and semicontiguous sequential patterns. In this configuration, if a pattern occurs more than 4 times out of 10 with a wildcard, it is considered semi-contiguous and thus not relevant ( 1 pattern). The 0.65 -closedness constraint allows to remove sub-patterns having a support 35 percent higher than a super-pattern and thus to remove more patterns than the closedness constraint (1 more pattern).
We have here an illustration of the capacity of the robustness constraint and the extended-closedness constraint to reduce the resulting.

The Table 8 is a summary of the various frequent sequential patterns with min_sup $=2$, which we have presented throughout this section.

### 5.4. The C3Ro Algorithm.

In this section we present the C3Ro algorithm (Algorithm 1), which integrates the framework that searches frequent $k$-contiguous $\delta$-robust sequential patterns satisfying a set of prefix-monotone constraints $\zeta$ (section 5.1), before checking the $\lambda$-closedness (section 5.2).

The input parameters of C3Ro are $S D B$, a sequence database, min_sup a minimum support, $\zeta$ a set of prefixmonotones constraints, $\delta$ a robustness ratio, $\lambda$ an extended ratio and $k$ a number of wildcards. C3Ro starts with an evaluation of the $k$-support of each item to extract $F 1$, the set of frequent items satisfying the set of constraints $\zeta$, according to the rules introduced in [44], via the function Enumeration items (line 2). The Backscan function (line 4) checks if $f 1$ can be

```
Algorithm \(1 \mathrm{C} 3 \operatorname{Ro}(S D B\), min_sup \(, \zeta, \delta, \lambda, k)\)
Input: \(S D B\) a sequence database, min_sup a minimum support, \(\zeta\) a set of prefix-monotones constraints, \(\delta\) a robustness ratio, \(\lambda\) an extended ratio and \(k\) a number of wildcards.
Output: \(F\), the \(C l_{\lambda} C o o_{k} R_{\delta} C_{\zeta} S P\)
```

```
\(F=\{\emptyset\}\)
```

$F=\{\emptyset\}$
$F 1=$ Enumeration_items $(S D B$, min_sup,$\zeta)$;
$F 1=$ Enumeration_items $(S D B$, min_sup,$\zeta)$;
for each $f 1$ in $F 1$ do
for each $f 1$ in $F 1$ do
if Backscan $(f 1)$
if Backscan $(f 1)$
$f 1-S D B=\mathbf{P}$ kdatabase $(S D B, f 1, k)$;
$f 1-S D B=\mathbf{P}$ kdatabase $(S D B, f 1, k)$;
$\mathbf{p E x t}\left(f 1 \_S D B, f 1\right.$,min_sup, $\left.\zeta, \delta, \lambda, k,-1, F\right) ;$
$\mathbf{p E x t}\left(f 1 \_S D B, f 1\right.$,min_sup, $\left.\zeta, \delta, \lambda, k,-1, F\right) ;$
return $F$;

```
return \(F\);
```

```
Algorithm \(2 \operatorname{pExt}\left(P_{-k} S D B, P\right.\), min_sup, \(\zeta, \delta, \lambda, k\),
cl_sup, F)
```

Input: $P_{-k} S D B$ the projected $k$-database of $k$-prefix $P$, $P$ a $k$-prefix, min_sup a minimum support, $\zeta$ a set of prefix-monotones constraints, $\delta$ a robustness ratio, $\lambda$ an extended ratio and $k$ a number of wildcards, $c l \_s u p$ the support the previous $C l S P$ and $F$ the previous set of $C l_{\lambda} C o_{k} R_{\delta} C_{\zeta} S P$.
Output: $F$, the current set of $C l_{\lambda} C o_{k} R_{\delta} C_{\zeta} S P$

```
\(F I=\) Enumeration \(\left(P \_S D B\right.\), min_sup \(\left., \zeta, \delta, k\right)\);
\(B E=\operatorname{Backward}\) extension \((P, k)\)
\(F E=\left|\left\{e \in F I \mid \sup _{A W_{k}}^{P S D B}(e)=\sup ^{S D B}(P)\right\}\right| ;\)
if \((\{B E \cup F E\}==\{\emptyset\})\)
    if \(c l \_\)sup \(==-1\)
            cl_sup \(=\sup _{A W_{k}}^{S D B}(P)\);
    \(\lambda B E=\lambda\)-Backward_extension \((P, k)\)
    \(\lambda F E=\left|\left\{e \in F I \mid \sup _{A W_{k}}^{P S D B}(e) \geq c l_{-} \sup * \lambda\right\}\right| ;\)
    if \((\{\lambda B E \cup \lambda F E\}==\{\emptyset\})\)
            cl_sup \(=-1\);
            if Check_C \((P, \zeta)\)
                \(F=F \cup\{P\} ;\)
for each \(i\) in \(F I\) do
    \(P_{i}=\langle P \diamond i\rangle ;\)
    if Backscan \(\left(P_{i}\right)\)
            \(P_{i} S D B=\mathbf{P}_{-}\)kdatabase \(\left(P_{-} S D B, P_{i}, k\right)\);
            \(\mathbf{p E x t}\left(P_{i} S D B, P_{i}\right.\), min_sup \(\left., \zeta, \delta, \lambda, k, c l_{-} s u p, F\right)\);
```

pruned according to Theorem 5.11. The projected $k$ - ${ }_{1416}$ database of each item in $F 1$ is created with the function ${ }_{1417}$ Pkdatabase (line 5). Then, the sub-algorithm pExt is ${ }_{1418}$ called with each frequent items satisfying all the prefix- 1419 monotonic constraints $\zeta$ (line 5), its goal is the recursive ${ }_{1420}$ path of the projected $k$-databases of each $k$-prefix to ex- ${ }^{1421}$ tract the complete set of patterns $C l_{\lambda} C o_{k} R_{\delta} C_{\zeta}$. In the ${ }_{1422}$ sub-algorithm pExt, the function Enumeration allows ${ }^{1423}$ the enumeration of the $\mathrm{Co}_{k} R_{\delta} C_{\zeta}$ patterns in the pro- ${ }_{1424}$ jected $k$-database of the $k$-prefix $P$ (line 1). The function ${ }_{1425}$ Backward extension obtains the $B E$ set of backward- ${ }_{1426}$ extensions of a the $k$-prefix $P$ according to the lemma ${ }_{1427}$ 5.9 (line 2). $F E$ is the set of forward-extensions of the $k-{ }_{1428}$ prefix $P$ obtained according to the lemma 5.6 (line 3). If ${ }_{1429}$ sets $B E$ and $F E$ are empty, then the $k$-prefix $P$ is closed ${ }_{1430}$ and its $k$-support must therefore be kept for the check- ${ }_{1431}$ ing of the $\lambda$-closedness constraint (lines $4,5,6$ ). In the ${ }_{1432}$ case where the $k$-prefix $P$ is closed, the $\lambda$-backward- ${ }_{1433}$ extensions and $\lambda$-forward-extensions are searched according to the lemmas 5.10 and 5.7 (lines 9,10 ). If the $k$-prefix has no extension and satisfies all the constraints ${ }^{1434}$ $\zeta$ (line 11), then it is added to the $F$ set of $C l_{\lambda} C o_{k} R_{\delta} C_{\zeta}$ (line 12) patterns. The sub-algorithm pExt is called re- ${ }^{1435}$ cursively (line 16) with a new $k$-prefix: the $P$ pattern ${ }^{1436}$ extended with each item in the $F I$ set, created during ${ }_{1437}$ the enumeration phase (line 1 ). When all $k$-prefixes ${ }^{1438}$ have been extended, the $F$ set is the complete set of ${ }_{1439}$ $C l_{\lambda} C o_{k} R_{\delta} C_{\zeta}$. C3Ro is thus an answer to the second scientific ques- ${ }^{1441}$ tion: How to design a sequential pattern mining algo- ${ }^{1442}$ rithm that aggregates several constraints such as mono- ${ }^{1443}$ tonic, anti-monotonic, closedness and contiguity con- 1444 straints and thus can be used with several combinations 1445 of constraints, whatever the needs of a user are?

The review of the literature of sequential pattern min- 1450 ing highlighted that mining large and noisy databases 1451 remains an important issue although such databases are ${ }_{1452}$ common in many practitioners' application contexts. ${ }^{1453}$ Using constraints during the mining process is a promis- ${ }^{1454}$ ing direction to alleviate this issue as they improve both 1455 the apprehensibility of the set of patterns and the effi- ${ }^{1456}$ ciency of the mining process. However, no algorithm ${ }_{1457}$ in the literature integrates constraints dedicated to noisy ${ }_{1458}$ databases.
C3Ro alleviates these limits. It is a generic sequential ${ }_{1460}$ pattern mining algorithm, able to mine both large and ${ }_{1461}$ noisy databases. It relies on two newly defined con- ${ }^{1462}$ straints and on the use of wildcards. Given that mem- ${ }^{1463}$ ory is the most limiting resource when mining large ${ }_{1464}$
databases, C3Ro is designed to limit memory usage. As a consequence, C3Ro may be less efficient in terms of execution time in comparison to recent algorithms that use vertical IDLists representation.
Furthermore, C3Ro is designed to be fully practitioneroriented, and uses understandable parameters that make sense in practitioners' application context. Nevertheless, these parameters are numerical parameters that need to be set by the user. To ensure that the best set of patterns is mined, the practitioner might have to experiment several combinations of values.
In conclusion, C3Ro does not guarantee to be the most efficient mining algorithm for each individual case, especially in terms of execution time. However, it is a generic algorithm, designed to be the only input (algorithm) that a practitioner uses, whatever are his/her needs, constraints and data characteristics. C3Ro fulfils a wide range of needs and is extremely practical.

## 7. Experiments

In this section, we conduct a comprehensive performance study to evaluate C3Ro in terms of efficiency, noise resistance and apprehensibility.

First, we aim at evaluating the efficiency of C3Ro when mining $\mathrm{ClCo}_{k} S P$ (closed $k$-contiguous sequential patterns) in terms of execution time, memory usage and scalability. As C3Ro is able to mine from closed contiguous sequential patterns ( $\operatorname{ClCoS} P$ ) to closed regular sequential patterns $(C l S P)$, depending on the value of $k$ (from 0 to $\infty$ ), we start by comparing C3Ro to CCSpan [58], a ClCoSP mining algorithm, and to CM-Clasp [13], one of the most efficient vertical ClS $P$ mining algorithm, in terms of both execution time and memory usage. Then, we evaluate the impact of each additional wildcard both on the execution time and on the number of mined patterns by C3Ro. For clarity, in this experiment, we use the notation $\mathrm{C}_{k} \mathrm{Ro}$ when C 3 Ro is in the configuration of mining $\mathrm{ClCo}_{k} S P$. The evaluation of the efficiency of $\mathrm{C} 3_{k}$ Ro is further detailed by an evaluation of its scalability on several datasets, including very large datasets.

Second, C3Ro is evaluated in terms of noise resistance and apprehensibility of the set of patterns, when mining $\mathrm{Cl}_{\lambda} \mathrm{Co}_{k} R_{\delta} C_{\zeta}$ ( $\lambda$-closed $k$-contiguous $\delta$-robust sequential patterns satisfying a set $\zeta$ of prefix-monotones constraints). As the employment sector is our main application field of interest, we conduct this second experiment on a dataset made up of job offers.

### 7.1. Datasets and Environment.

The experiments are performed on a i7-7700HQ ${ }_{1496}$ 2.8 GHz with 16 GB memory on Windows 10.

In order to evaluate the efficiency of C3Ro (first part ${ }^{1498}$ of the experiments), we use three reference datasets 1499 (see Table 9), traditionally used to evaluate sequen- 1500 tial pattern mining algorithms [51, 47, 13, 58, 2]. 1501 BMS 1 Gazelle, Kosarak and a light version LKosarak. 1502 BMS 1 Gazelle and Kosarak are available online on the ${ }_{1503}$ SPMF website ${ }^{1}$.
The BMS 1 Gazelle dataset contains clickstream and 1505 purchase data from Gazelle.com, a legwear and legcare 1506 web retailer, it was used in the KDD-CUP $2000{ }_{1507}$ competition. The second dataset, Kosarak, is a very ${ }_{1508}$ large dataset containing almost 1 million sequences 1509 of clickstream data from a Hungarian news portal. 1510 The light version (LKosarak) contains about a third ${ }_{1511}$ of the sequences of the original, with almost the same ${ }_{1512}$ characteristics.
The second part of the experiments is conducted on 1514 Jobs 60 K [2]. This dataset is made up of 60,000 job 1515 offers, collected from specialized websites between ${ }_{1516}$ January 2016 and June 2016. Jobs $60 K$ is noisy and 1517 very sparse, it reflects the kind of textual data that can ${ }_{1518}$ be found on the Web.


Figure 2: Execution time on Kosarak

### 7.2. Efficiency Evaluation

### 7.2.1. C3Ro vs CM-Clasp vs CCSpan

This section is interested in the execution time (Fig- ${ }^{1543}$ ures 1 and 2 ) and the memory usage (Table 10) of the $3{ }^{1544}$

[^1]algorithms on the 3 datasets.
We first focus on the comparison of $C 3_{0}$ Ro and CCSpan, as they both mine $\operatorname{ClCoS} P$, in terms of execution time, with various relative support thresholds, on BMS 1 Gazelle. The execution time of both CCSpan and $C 3_{0}$ Ro on BMS 1 Gazelle, according to min_sup, is shown in Figure 1a. $C 3_{0}$ Ro always has an execution time below CCSpan, up to a maximum of almost 700 times with the lowest support. In addition, the decrease of the support impacts twice less $C 3_{0} R o$ than CCSpan. In addition, even with $k$ ranging from 1 to $\infty$, $C 3_{k}$ Ro remains faster than CCSpan on BMS 1 Gazelle. The execution time of CCSpan on both Kosarak and LKosarak is not shown due to an extremely high execution time.

Second, we focus on the mining of ClS $P$ by comparing $C 3_{k} R o$ and CM-Clasp with a number of wildcards $k$ ranging from 0 to $\infty$ (Figures 1 and 2). CM-Clasp could not be executed on any of the 3 datasets with a support lower than $10^{-3}$, due to an extreme memory usage. We observe that when the number of wildcards is below 5 and the support is higher than $10^{-3}, C 3_{k} R o$ and CM-Clasp are very similar in terms of execution time on the three datasets. Therefore, we can conclude that, when $k$ is ranging from 0 to $5, C 3_{k}$ Ro competes with one of the most efficient $\operatorname{CloS} P$ mining algorithms, CM-Clasp, while being able to run with very low support values on very large datasets such as Kosarak. Let us recall that C3Ro has been designed to improve the efficiency of sequential pattern mining and the apprehensibility of its results. Thus, even if C3Ro can mine $C l S P$, it is rather intended to mine contiguous and semi-contiguous sequential patterns, so with a limited number of wildcards, in order to achieve its objective regarding efficiency and apprehensibility.

Regarding the memory usage (Table 10), let us first focus on the comparison of CCSpan and $C 3_{0}$ Ro that mine ClCoS P. On BMS 1 Gazelle, they are both consuming less than 300 MB , so are comparable. Let us recall that CCSpan could not be executed on both LKosarak and Kosarak due to execution time issues.
Let us now focus on the comparison of CM-Clasp and $C 3_{\infty}$ Ro that mine CloSP. On BMS 1 Gazelle, CM-Clasp consumes at least $4,200 \mathrm{MB}$ and explodes regarding memory usage when is min_sup lower than $5.10^{-3} . C 3_{\infty}$ Ro consumes less than 250 MB , whatever is min_sup, which is similar to $C 3_{0} R o$.

Concerning LKosarak and Kosarak, we observe a similar behavior. $C 3_{\infty}$ Ro has a low memory usage, i.e. between 500 MB and $1,300 \mathrm{MB}$ according to min_sup

(a) min_sup (in \%)

(b) min_sup (in \%)

Figure 1: Execution time on Gazelle(a) and LKosarak(b)
and CM-Clasp has a very high memory usage, between 1574 $6,700 \mathrm{MB}$ and $10,600 \mathrm{MB}$ with a high min_sup value, ${ }_{1575}$ and explodes when the support is lower than $2.10^{-2}$. ${ }_{1576}$ This high memory usage is a drawback of the verti- ${ }_{1577}$ cal representation when the number of frequent patterns ${ }_{1578}$ mined increases. This makes CM-Clasp not executable ${ }_{1579}$ on any dataset with a low min_sup value. We observe ${ }_{1580}$ that the number of wildcards has no impact on the memory usage of $C 3_{k} R o$, probably due to the pattern growth representation.

With a limited number of wildcards, $C 3_{k} R o$ has an execution time similar to CM-Clasp and becomes slower when the number of wildcards is above 5 . Let us recall that, even if $C 3_{k} R o$ is able to mine $C l S P$, it has been designed to mine closed contiguous or semi-contiguous sequential patterns, therefore the number of wildcards is not supposed to be high. In the frame of $\operatorname{ClCoS} P$ mining, $C 3_{0}$ Ro is up to 700 times faster than CCSpan. At ${ }^{1592}$ last, $C 3_{k}$ Ro has a very low memory usage, contrary to ${ }^{1593}$ CM-Clasp that cannot use a low min_sup value, due an ${ }^{1594}$ excessive memory usage of the vertical IDLists repre- ${ }^{1595}$ sentation.

We can conclude that among CCSpan, CM-Clasp and 1599 $C 3_{k} R o, C 3_{k} R o$ is the only sequential pattern mining al- 1600 gorithm able to mine large datasets in a reasonable time, 1601 including with low min_sup values, due to its low mem- 1602 ory usage.

### 7.2.2. Impact of the number of wildcards

Let us now focus on the impact of the number of wildcards on $C 3_{k} R o$ execution time, on the 3 datasets (Figures 1 and 2). We also propose to study the impact of the number of wildcards on the size of the set of mined patterns.

First, we focus on the configuration where the highest support is used. On the BMS 1 Gazelle dataset, each additional wildcard increases the execution time by $2 \%$ on average. On the LKosarak and Kosarak datasets, this increase is higher, on average $7 \%$ and $11 \%$ respectively.
When focusing on the associated number of mined patterns, it is unexpectedly slightly impacted. This small impact may be due to the support value that may be too high.

Second, we focus on the configuration where the lowest support is used. In terms of execution time, unlike in the previous configuration, each wildcard has a significant impact. Indeed, regardless of the rank of the wildcard, the impact on the execution time is about $30 \%$ on BMS 1 Gazelle and $100 \%$ on Kosarak. In terms of number of patterns, at the opposite of the previous configuration, each wildcard allows to mine a significant additional number of patterns (on average twice more), whatever is the dataset used.

We can conclude that the impact of the use of one
wildcard, on both execution time and number of pat- ${ }_{1655}$ terns, highly depends on its rank, the minimum support ${ }_{1656}$ value and the dataset characteristics. The largest impact ${ }_{1657}$ is achieved when the lowest support is used. This im- ${ }_{1658}$ pact is explained by the large number of additional pat- ${ }_{1659}$ terns provided by each wildcard. This result supports 1660 the fact that the number of wildcards is a critical pa- ${ }^{1661}$ rameter that greatly impacts the execution time and the ${ }_{1662}$ number of mined patterns.

We now propose to evaluate the scalability of $C 3_{k} R o$, 1666 through the use of various number of sequences in ${ }_{1667}$ datasets. We use both LKosarak and Kosarak, which 1668 have similar characteristics, except the number of 1669 sequences that is three times smaller in LKosarak. ${ }_{1670}$ With the lowest support, which represents the con- 1671 figuration where the execution time and the number ${ }_{1672}$ of mined patterns are the highest, $C 3_{0}$ Ro runs in $22{ }_{1673}$ seconds on LKosarak and in less than 80 seconds on 1674 Kosarak (4 times faster on LKosarak). It is worth 1675 noting that CCSpan (which mines the same patterns ${ }_{1676}$ than $C 3_{0} R o$ ) cannot be executed in a decent amount of time on both datasets, whatever is the minimum support 1677 value used. $C 3_{5}$ Ro runs 10 times faster on LKosarak ${ }_{1678}$ (200 seconds) than on Kosarak ( 2,000 seconds ). We 1679 can say that the use of wildcards only slightly impacts ${ }_{1680}$ the scalability of C3Ro. Let us recall that CM-Clasp, 1681 one of the most efficient CloSP mining algorithm, 1682 cannot be executed on any of the three datasets when ${ }_{1683}$ the support is under $5 \cdot 10^{-3}$. Nevertheless, $C 3_{\infty} R o{ }_{1684}$ can be executed with a very low memory usage with 1685 such low supports. These elements highlight the high ${ }_{1686}$ scalability of $C 3_{k} R o$.

Two more arguments are put forward to illustrate the 1689 scalability of $C 3_{k} R o$ :
To the best of our knowledge, no sequential pattern ${ }_{1691}$ mining algorithms in the literature can be executed on 1692 both LKosarak and Kosarak, in a decent amount of 1693 time, when the support is lower than $5.10^{-3}$. The fact ${ }_{1694}$ that $C 3_{0}$ Ro and $C 3_{7}$ Ro can be executed in a reasonable ${ }_{1695}$ time on Kosarak with such a low support, is a first 1696 indicator of its scalability.
At last, the configuration of C3Ro used was dedicated ${ }_{1698}$ to the mining of $C l_{1} C o_{k} R o_{1} C_{\emptyset} S P$, for the sake of ${ }_{1699}$ comparison with CCSpan and CM-Clasp. If different 1700 values of $\lambda$ and $\delta$ are used, the computation time will be 1701 even more lower. We can thus conclude that the mining ${ }_{1702}$ of $C l_{\lambda} C o_{k} R o_{\delta} C_{\zeta} S P$ does not impact the scalability of ${ }_{1703}$ C3Ro in terms of execution time.

To summarize, this first set of experiments has shown that C3Ro is always significantly faster than CCSpan and similar to CM-Clasp, when mining closed contiguous or semi-contiguous sequential patterns. Moreover, the number of wildcards has an impact on the execution time of C3Ro, with an average increase of $33 \%$ per wildcard on each of the three datasets. On these three datasets, the use of wildcards only slightly impacts the scalability. Furthermore, this increase in the execution time was expected due to the large number of additional patterns mined when using wildcards ( $40 \%$ for each wildcard) on the three datasets.
This experiment has also shown that C3Ro has a very low memory usage regardless of the support, the dataset or the number of wildcards, while the memory usage is the main flaw of CM-Clasp due to the vertical IDLists representation.

C3Ro has thus proven to be a scalable algorithm with a really small execution time on a very large datasets, such as Kosarak, especially in comparison to CCSpan and CM-Clasp, thus to closed sequential pattern mining algorithms.

### 7.3. Noise resistance of C3Ro and apprehensibility of

 the set of patternsIn this section, we show how the extended-closedness and the robustness constraints that we have introduces impact the number and the relevance of the patterns mined in noisy data, especially in a contiguous sequential pattern mining frame, so the apprehensibility of the set of patterns.
This experiment is conducted on the Jobs 60 K dataset, which contains 60 K job offers with an average length of 77.0 on 92,394 distinct items (see Table 9).

In the employment sector, activities are at the core of job offers. An activity is a coherent set of completed tasks, organized toward a predefined objective with a result that can be measured. An activity is usually formalized by action verbs [1]. In this context, we define activities as patterns starting with a verb and with a length no less than 3 . We view these characteristics as constraints. Thus, we set two prefix-monotone constraints: $C_{1}$ as a pattern having a first item of the verb category $(V)$ : $C_{1}(P)=P[1] \in V$, and $C_{2}$ on the length of the pattern that should be no less than 3: $C_{2}(P)=\operatorname{len}(P)>2$. "Inform clients by explaining procedures", "Manage business by performing related duties" or "Start operations by entering commands" are examples of activities. Experts agree that the length of an activity is never less than 3 and is on average made up of between 3 and 6 words.
We set a relative support of $15.10^{-4} \%$ in order to mine

Table 11: $C l_{0.8} C o_{1} R_{0.5} C_{\left\{C_{1}, C_{2}\right\}} S P$ in Jobs $60 K$

| id | configuration | activities |
| :--- | :--- | ---: |
| $\# 1$ | $C l_{1} C o o_{0} R_{1} C_{\left\{C_{1}, C_{2}\right\}}$ | 352 |
| $\# 2$ | $C l_{1} C o o_{1} R_{1} C_{\left\{C_{1}, C_{2}\right\}}$ | 698 |
| $\# 3$ | $C l_{1} C o o_{1} R_{0.5} C_{\left\{C_{1}, C_{2}\right\}}$ | 429 |
| $\# 4$ | $C l_{0.8} C o l_{1} R_{1} C_{\left\{C_{1}, C_{2}\right\}}$ | 601 |
| $\# 5$ | $C l_{0.8} C o o_{1} R_{0.5} C_{\left\{C_{1}, C_{2}\right\}}$ | 384 |

a significant amount of patterns and perform a relevant ${ }^{1757}$ analysis. An experiment conducted in [2] on the same ${ }_{1759}$ dataset has shown that $k=1$ wildcard is the adequate ${ }_{1760}$ number to mine activities. Indeed, when more than one ${ }_{1761}$ wildcard are allowed, the patterns mined become too ${ }_{1762}$ long (average length above 11) and are therefore often ${ }_{1763}$ meaningless.
We will not study the impact of the variation of the val- ${ }_{1765}$ ues of $\delta$ and $\lambda$, as it would only give information on the ${ }_{1766}$ dataset regarding activities. We set the robustness ra- ${ }_{1767}$ tio to $\delta=0.5$ and the extended ratio to $\lambda=0.8$. These ${ }_{1768}$ values have been determined experimentally during sev- ${ }_{1769}$ eral runs to reduce the impact of the noise on the mined 1770 activities, while preventing at best the mining of irrel- ${ }_{1771}$ evant activities in the final set. For example, $\delta=0.5{ }_{1772}$ means that an activity occurring with a wildcard more ${ }_{1773}$ than $50 \%$ of the cases does not represent a real activity. ${ }_{1774}$

Let us first focus on the number of mined activities 1776 according to the parameters of C3Ro (table 11): with ${ }_{1777}$ a wildcard $(k=1)$ or without any $(k=0)$, with a ${ }_{1778}$ robustness ratio $(\delta=0.5)$ and without any $(\delta=1)$, with 1779 an extended ratio $(\lambda=0.8)$ or without any $(\lambda=1)$. We ${ }_{1880}$ refer to the different configurations of C3Ro by using ${ }_{1781}$ the id indicated in Table 11. We start by evaluating the 1782 impact of the use of $k=1$ wildcard on the number of ${ }_{1783}$ mined activities by comparing the results of \#1 and \#2. ${ }^{1784}$ \#1 mines 352 activities, while \#2 mines 698 activities. ${ }^{1785}$ The use of one wildcard allows to find 346 additional ${ }_{1786}$ activities, which doubles the number of activities. This ${ }_{1787}$ increase suggests that the dataset could be noisy and ${ }_{1788}$ that the use of a wildcard is relevant. However, this 1789 configuration cannot differentiate activities that are ${ }_{1790}$ frequently contiguous, thus that were not mined by \#1 1791 due the noise, from the activities that are frequently not 1792 contiguous, thus artificially created by the wildcard. ${ }_{1793}$ Thus, this configuration does not give any information 1794 about the relevance of the newly mined activities. 1795 To tackle this issue, \#3 mines 429 activities, meaning 1796 sequential patterns that are not relevant in activity mining and the 0.8 -closedness removes activities included in other "super-activity" with a similar support ( $20 \%$ lower at maximum). Therefore, the combination of both new constraints allows to improve even more the apprehensibility of the resulting set.
The comparison between the set of mined activities by \#5 and the one mined by \#1 shows that 280 activities are common to both sets. We now focus on the evaluation of the relevancy of the activities specific to each set. The 72 remaining activities of the set mined by \#1 were assessed by an expert of the field, who concluded that more than $50 \%$ of these 72 activities are either irrelevant or redundant. At the opposite, the analysis, by the same expert, of the 104 activities specific to \#5, highlighted that only $10 \%$ of these activities are either irrelevant or redundant. This analysis confirms that 0.8 -closed 1 -contiguous 0.5 -robust sequential patterns allow to reduce the final set of patterns by removing useless patterns, while increasing the relevance of the patterns mined. In a nutshell, \#5 allows to mine around $20 \%$ more relevant activities than \#1 in noisy data, while only increasing by $9 \%$ the resulting set size and thus improve the apprehensibility in the mining of activities.

To conclude, this experiment shows that the use of ${ }_{1846}$ the two newly introduced constraints (robustness and 1847 extended-closed) is a lever to improve the apprehensi- 1848 bility of the results, with no impact on the execution 1849 time. In addition, the use of these new constraints in 1850 C3Ro is flexible through the use of a robustness and an 1851 extended ratios allowing C3Ro to be adjustable to the ${ }_{1852}$ final user needs.
C3Ro is thus an answer to the first scientific question: ${ }_{1854}$ How to efficiently mine an apprehensible set of sequen- ${ }_{1855}$ tial patterns, including the frame of noisy data?

### 7.4. Expert feedback

The mining of activities in job offers is a key component of the job market's analysis. Indeed, the mined activities provide a relevant picture of the needs of companies, which is essential in many areas such as consulting, recruitment and training. In consulting, it provides relevant factual elements to support companies in expressing their own needs in terms of recruitment.

Table 12 illustrates how the use of C3Ro significantly improves the set of mined activities in Jobs 60 K . The activity "managing sales teams" is no longer mined with C3Ro as it is included in the activity "managing sales teams effectively", which is $12 \%$ more frequent. In that pattern, the word "effectively" is as important as the rest of the activity. This fact has been validated by a HR expert. In addition, C3Ro discovers a new activity "managing sales teams distribution channels", which was not frequent with traditional algorithms, and is now discovered thanks to the use of the two new constraints. This activity has also been validated by the HR expert. More generally, the set of mined activities is more apprehensible. Not only its size is decreased but the information it contains is more valuable.

## 8. Conclusion

While sequential pattern mining has been widely ${ }_{1885}$ studied for many years, it still faces several challenges. ${ }^{1886}$ First, many studies have focused on increasing the ef- ${ }^{1887}$ ficiency of the mining process and on decreasing the ${ }_{1888}$ size of the set of patterns, but mining large databases re- 1889 mains a challenge. Second, although noise is a common 1890 feature in real-world databases, it has not faced much 1891 interest in the pattern mining literature. Third, the re- 1892 duction of the size of the set of patterns is often at the ${ }_{1893}$ expense of an uncontrolled loss of information, which 1894 may make the set of patterns useless, especially when it ${ }_{1895}$ is intended to practitioners.

 designed to mine large and noisy databases, while discovering an apprehensible set of patterns, i.e. a set of patterns of reduced size, with a controlled information loss.

To reach this goal, we introduced two new constraints, namely the robustness constraint that allows the mining of contiguous sequential patterns in noisy data and the extended-closedness constraint, which represents a trade-off between the full information extracted with the closedness constraint and the reduced set mined with the maximal constraint. In addition, we proposed a generic algorithm C3Ro, based on the pattern-growth philosophy that manages several parameters. These parameters have been chosen to mine a large range of patterns: going from closed contiguous sequential patterns to maximal regular sequential patterns. C3Ro is thus thought to be the single pattern mining algorithm that can fit the final users' specific needs, whatever they are, thanks to these parameters
The experiments we conducted showed that C3Ro is scalable with a very low memory usage compared to well-known algorithms such as CCSpan and CM-Clasp. An expert has validated that the use of the two new constraints allows to mine more relevant patterns compared to the use of the standard contiguity or closedness constraints in noisy data.

Based on this work, we intend to work on four future directions. First, although C3Ro is designed for practitioners' use, the required parameters need to be fixed by these practitioners, especially those associated to both constraints ( $\delta$ and $\lambda$ ). Their optimal value can only be found by iteratively running C3Ro. We would like to work on the automatic determination of these values, at least of a limited range of possible values. It will rely on an analysis of the type and amount on noise in data. Limiting this possible range of values will be of high utility for practitioners.
Second, still with the objective of satisfying practitioners, we plan to focus on new quality criteria. Indeed, support is the traditionally used as the evaluation criterion, but unexpectedness, periodicity, coherence, etc. also deserve to be studied for mining patterns that may fit better practitioners' expectations.
Third, we intend to focus on noise, especially on studying the precise position of noise within patterns (in what places does noise occur). Not only the identification of this noise will help quantify and qualify it, but its management will contribute to represent patterns more pre-
cisely and thus increase even more the impact of both 1959 new constraints on apprehensibility.
Last, we will use the output of the third goal as a first ${ }_{1962}^{1961}$ base to study pattern drift. We view the management ${ }_{1963}$ of the position of noise as a possibility to differentiate ${ }^{1964}$ between noise and new forms of patterns (loss, substi- 1965 tution or addition of items in existing patterns). This last ${ }^{1960}$ goal is of high utility for many real-world applications, ${ }_{1968}$ as it is common that data evolve and this evolution has ${ }^{1969}$ to be managed.

1] Abboud, Y., Boyer, A., and Brun, A. (2015). Predict the emer- ${ }^{1972}$ gence: Application to competencies in job offers. In Tools with ${ }^{19}$ Artificial Intelligence (ICTAI), 2015 IEEE 27th International Con- 1974 ference on, pages 612-619. IEEE.
[2] Abboud, Y., Boyer, A., and Brun, A. (2017). CCPM: A Scal- ${ }^{1976}$ able and Noise-Resistant Closed Contiguous Sequential Patterns Mining Algorithm. In 13th International Conference on Machine Learning and Data Mining MLDM 2017, volume 89, page 15, New York, United States.
[3] Aggarwal, C. C., Li, Y., Wang, J., and Wang, J. (2009). Frequent pattern mining with uncertain data. In Proceedings of the 15 th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 29-38. ACM.
[4] Aggarwal, C. C., Ta, N., Wang, J., Feng, J., and Zaki, M. (2007). Xproj: a framework for projected structural clustering of xml documents. In Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 4655. ACM.
[5] Agrawal, R., Imieliński, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In Acm sigmod record, volume 22, pages 207-216. ACM.
[6] Agrawal, R. and Srikant, R. (1995). Mining sequential patterns. In Data Engineering, 1995. Proceedings of the Eleventh International Conference on, pages 3-14. IEEE
[7] Ayres, J., Flannick, J., Gehrke, J., and Yiu, T. (2002). Sequential pattern mining using a bitmap representation. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’02, pages 429-435, New York, NY, USA. ACM.
[8] Béchet, N., Cellier, P., Charnois, T., and Crémilleux, B. (2015). ${ }^{2001}$ Sequence mining under multiple constraints. In Proceedings of 2002 the 30th Annual ACM Symposium on Applied Computing, pages 2003 908-914. ACM.
[9] Chen, J. (2008). Contiguous item sequential pattern mining using ${ }^{2005}$ updown tree. Intelligent Data Analysis, 12(1):25-49.
[10] Chen, J. and Cook, T. (2007). Mining contiguous sequential ${ }^{2007}$ patterns from web logs. In Proceedings of the 16th International ${ }^{2008}$ Conference on World Wide Web, WWW '07, pages 1177-1178, ${ }_{2009} 2009$ New York, NY, USA. ACM.
[11] Djenouri, Y., Belhadi, A., and Fournier-Viger, P. (2017). Ex- 2011 tracting useful knowledge from event logs: A frequent itemset ${ }^{2012}$ mining approach. Knowledge-Based Systems.
[12] Fischer, J., Heun, V., and Kramer, S. (2006). Optimal string mining under frequency constraints. In $P K D D$, volume 4213, pages ${ }^{2}$ 139-150. Springer.
[13] Fournier-Viger, P., Gomariz, A., Campos, M., and Thomas, R. ${ }^{2017}$ (2014a). Fast vertical mining of sequential patterns using co- 2018 occurrence information. In Pacific-Asia Conference on Knowledge ${ }^{20}$ Discovery and Data Mining, pages 40-52. Springer.
[14] Fournier-Viger, P., Gomariz, A., Šebek, M., and Hlosta, M. ${ }^{2021}$ (2014b). Vgen: fast vertical mining of sequential generator pat- ${ }^{2022}$ terns. In International Conference on Data Warehousing and ${ }^{2023}$

Knowledge Discovery, pages 476-488. Springer.
[15] Fournier-Viger, P., Nkambou, R., and Nguifo, E. M. (2008). A knowledge discovery framework for learning task models from user interactions in intelligent tutoring systems. In Mexican International Conference on Artificial Intelligence, pages 765-778. Springer.
[16] Fournier-Viger, P., Wu, C.-W., Gomariz, A., and Tseng, V. S. (2014c). Vmsp: Efficient vertical mining of maximal sequential patterns. In Canadian Conference on Artificial Intelligence, pages 83-94. Springer.
[17] Fumarola, F., Lanotte, P. F., Ceci, M., and Malerba, D. (2016). Clofast: closed sequential pattern mining using sparse and vertical id-lists. Knowledge and Information Systems, 48(2):429-463.
[18] Fürnkranz, J. (1998). A study using n-gram features for text categorization. Austrian Research Institute for Artifical Intelligence, 3(1998):1-10.
[19] Gao, C., Wang, J., He, Y., and Zhou, L. (2008). Efficient mining of frequent sequence generators. In Proceedings of the 17th international conference on World Wide Web, pages 1051-1052. ACM.
[20] García, S., Luengo, J., and Herrera, F. (2015). Data preprocessing in data mining. Springer.
[21] García-Hernández, R. A., Martínez-Trinidad, J. F., and Carrasco-Ochoa, J. A. (2006). A new algorithm for fast discovery of maximal sequential patterns in a document collection. In CICLing, pages 514-523. Springer.
[22] Garofalakis, M. N., Rastogi, R., and Shim, K. (1999). SPIRIT: Sequential pattern mining with regular expression constraints. In Proceedings of the 25th International Conference on Very Large Data Bases, VLDB '99, pages 223-234, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
[23] Gomariz, A., Campos, M., Marin, R., and Goethals, B. (2013). Clasp: an efficient algorithm for mining frequent closed sequences. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 50-61. Springer.
[24] Grossi, R., Iliopoulos, C. S., Liu, C., Pisanti, N., Pissis, S. P., Retha, A., Rosone, G., Vayani, F., and Versari, L. (2017). On-line pattern matching on similar texts. In LIPIcs-Leibniz International Proceedings in Informatics, volume 78. Schloss Dagstuhl-LeibnizZentrum fuer Informatik.
[25] Hahsler, M. and Karpienko, R. (2017). Visualizing association rules in hierarchical groups. Journal of Business Economics, 87(3):317-335
[26] Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q., Dayal, U., and Hsu, M.-C. (2000). Freespan: frequent pattern-projected sequential pattern mining. In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 355-359. ACM.
[27] Han, J., Pei, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., and Hsu, M. (2001). Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In Proceedings of the 17th international Conference on Data Engineering, pages 215224.
[28] Hirate, Y. and Yamana, H. (2006). Generalized sequential pattern mining with item intervals. JCP, 1(3):51-60.
[29] Ho, J., Lukov, L., and Chawla, S. (2005). Sequential pattern mining with constraints on large protein databases. In Proceedings of the 12th International Conference on Management of Data (COMAD), pages 89-100.
[30] Kang, T. H., Yoo, J. S., and Kim, H. Y. (2007). Mining frequent contiguous sequence patterns in biological sequences. In 2007 IEEE 7th International Symposium on BioInformatics and BioEngineering, pages 723-728.
[31] Karim, M., Rashid, M., Jeong, B.-S., Choi, H.-J., et al. (2012). An efficient approach to mining maximal contiguous frequent pat-
terns from large dna sequence databases. Genomics $\mathcal{E}$ informatics, 2089 10(1):51-57.
[32] Kemmar, A., Loudni, S., Lebbah, Y., Boizumault, P., and 2091 Charnois, T. (2015). Prefix-projection global constraint for sequen- 2092 tial pattern mining. In International Conference on Principles and 2093 Practice of Constraint Programming, pages 226-243. Springer. 2094
[33] Le, B., Duong, H., Truong, T., and Fournier-Viger, P. (2017). 2095 Fclosm, fgensm: two efficient algorithms for mining frequent 2096 closed and generator sequences using the local pruning strategy. 2097 Knowledge and Information Systems, pages 1-37.
[34] Li, C. and Wang, J. (2008). Efficiently mining closed subse- 2099 quences with gap constraints, pages 313-322.

2100
[35] Li, H., Zhang, N., Zhu, J., Wang, Y., and Cao, H. (2018). Proba- 2101 bilistic frequent itemset mining over uncertain data streams. Expert 2102 Systems with Applications.
[36] Liao, V. C.-C. and Chen, M.-S. (2014). DFSP: a depth-first 2104 spelling algorithm for sequential pattern mining of biological se- 2105 quences. Knowledge and information systems, 38(3):623. 2106
[37] Liu, H., Wang, L., Liu, Z., Zhao, P., and Wu, X. (2018). Efficient 2107 pattern matching with periodical wildcards in uncertain sequences. 2108 Intelligent Data Analysis, 22(4):829-842.

2109
[38] Liu, J., Paulsen, S., Wang, W., Nobel, A., and Prins, J. (2005). 2110 Mining approximate frequent itemsets from noisy data. In Data 2111 Mining, Fifth IEEE International Conference on, pages 4-pp. 2112 IEEE.
39] Luo, C. and Chung, S. M. (2005). Efficient mining of maximal 2114 sequential patterns using multiple samples. In Proceedings of the 2115 2005 SIAM International Conference on Data Mining, pages 415-2116 426. SIAM.
[40] Mannila, H., Toivonen, H., and Inkeri Verkamo, A. (1997). Dis- 2118 covery of frequent episodes in event sequences. Data Mining and 2119 Knowledge Discovery, 1(3):259-289.

2120
[41] Matsui, T., Uno, T., Umemori, J., and Koide, T. (2013). A New 2121 Approach to String Pattern Mining with Approximate Match, pages 2122 110-125. Springer Berlin Heidelberg, Berlin, Heidelberg.
[42] Pasquier, N., Bastide, Y., Taouil, R., and Lakhal, L. (1999). Discovering frequent closed itemsets for association rules. In Proceedings of the 7th International Conference on Database Theory, ICDT '99, pages 398-416, London, UK, UK. Springer-Verlag.
[43] Pei, J., Han, J., Mao, R., et al. (2000). CLOSET: An efficient algorithm for mining frequent closed itemsets. In ACM SIGMOD workshop on research issues in data mining and knowledge discovery, volume 4, pages 21-30.
[44] Pei, J., Han, J., and Wang, W. (2007). Constraint-based sequential pattern mining: the pattern-growth methods. Journal of Intelligent Information Systems, 28(2):133-160.
[45] Rodríguez-González, A. Y., Lezama, F., Iglesias-Alvarez, C. A., Martínez-Trinidad, J. F., Carrasco-Ochoa, J. A., and de Cote, E. M. (2017). Closed frequent similar pattern mining: Reducing the number of frequent similar patterns without information loss. Expert Systems with Applications.
[46] Srikant, R. and Agrawal, R. (1996). Mining sequential patterns: Generalizations and performance improvements. Advances in Database TechnologyEDBT'96, pages 1-17.
[47] Wang, J. and Han, J. (2004). BIDE: Efficient mining of frequent closed sequences. In Proceedings of the 20th International Conference on Data Engineering, ICDE '04, pages 79-, Washington, DC, USA. IEEE Computer Society.
[48] Wang, J., Han, J., and Pei, J. (2003). CLOSET+: Searching for the best strategies for mining frequent closed itemsets. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '03, pages 236245, New York, NY, USA. ACM.
[49] Wu, X., Zhu, X., He, Y., and Arslan, A. N. (2013). Pmbc: Pattern mining from biological sequences with wildcard constraints.

Computers in biology and medicine, 43(5):481-492.
[50] Xie, F., Wu, X., and Zhu, X. (2017). Efficient sequential pattern mining with wildcards for keyphrase extraction. Knowledge-Based Systems, 115(Supplement C):27-39.
[51] Yan, X., Han, J., and Afshar, R. (2003). Clospan: Mining: Closed sequential patterns in large datasets. In Proceedings of the 2003 SIAM international conference on data mining, pages 166177. SIAM.
[52] Yang, J., Wang, W., Yu, P. S., and Han, J. (2002). Mining long sequential patterns in a noisy environment. In Proceedings of the 2002 ACM SIGMOD international conference on Management of data, pages 406-417. ACM.
[53] Yi, S., Zhao, T., Zhang, Y., Ma, S., and Che, Z. (2011). An effective algorithm for mining sequential generators. Procedia Engineering, 15:3653-3657
[54] Yun, U. and Ryu, K. H. (2011). Approximate weighted frequent pattern mining with/without noisy environments. KnowledgeBased Systems, 24(1):73-82.
[55] Zaki, M. J. (2000). Sequence mining in categorical domains: Incorporating constraints. In Proceedings of the Ninth International Conference on Information and Knowledge Management, CIKM '00, pages 422-429, New York, NY, USA. ACM.
[56] Zaki, M. J. (2001). SPADE: An efficient algorithm for mining frequent sequences. Machine learning, 42(1):31-60.
[57] Zaki, M. J. and Hsiao, C.-J. (2002). CHARM: An efficient algorithm for closed itemset mining. In Proceedings of the 2002 SIAM international conference on data mining, pages 457-473. SIAM.
[58] Zhang, J., Wang, Y., and Yang, D. (2015). CCSpan: Mining closed contiguous sequential patterns. Knowledge-Based Systems, 89:1-13.
[59] Zhang, M., Kao, B., Cheung, D. W., and Yip, K. Y. (2007). Mining periodic patterns with gap requirement from sequences. ACM Transactions on Knowledge Discovery from Data (TKDD), 1(2):7.

Table 4：Abbreviations

| abbreviations | type of patterns |
| :--- | :--- |
| $S P$ | Sequential Pattern |
| $C l S P$ | Closed Sequential Pattern |
| $C o S P$ | Contiguous Sequential Pattern |
| $C l C o S P$ | Closed Contiguous Sequential Pattern |
| $C l C o{ }_{k} R_{\delta} S P$ | Closed $k$－Contiguous $\delta$－Robust Sequential Pattern |
| $C l_{\lambda} C o_{k} S P$ | $\lambda$－Closed $k$－Contiguous Sequential Pattern |
| $C l_{\lambda} C o_{k} R_{\delta} C_{\zeta}$ | $C l_{\lambda} C o_{k} R_{\delta}$ with a set of Constraints $\zeta$ |

Table 5：Type of sequential patterns mined in the Noise database

| type of patterns | patterns |
| :---: | :---: |
| CoS P | $\langle$ dear $\rangle: 3,\langle$ dear，Watson $\rangle: 2,\langle$ Elementary $\rangle: 5,\langle$ Elementary，my $\rangle: 5$ ， $\langle$ Elementary，my，dear $\rangle$ ：3，〈Elementary，my，dear，Watson $\rangle$ ：2， $\langle m y\rangle: 5,\langle m y$, dear $\rangle: 3,\langle m y$, dear，Watson $\rangle: 2$ |
| ClCoS P | $\langle$ Elementary，my ：5，〈Elementary，my，dear $\rangle$ ：3， <br> $\langle$ Elementary，my，dear，Watson $\rangle: 2,\langle$ Watson $\rangle: 5$ |
| $\mathrm{ClCo}_{1} \mathrm{SP}$ | $\begin{aligned} & \hline \hline \text { Elementary, my }\rangle: 5,\langle\text { Elementary, my, dear, Watson }\rangle: 3 \text {, } \\ & \langle\text { Elementary, my, Watson }\rangle: 4,\langle\text { Watson }\rangle: 5 \end{aligned}$ |
| $\mathrm{ClCo}_{1} \mathrm{R}_{0.6} \mathrm{SP}$ | $\langle$ Elementary，my ：5，〈Elementary，my，dear，Watson ：5，＜Watson $\rangle$ ： 5 |
| $C l_{0.6} C o_{1} R_{0.6} S P$ | 〈Elementary，my，dear，Watson ： 3 |

Table 8：Types of frequent sequential patterns in $S D B$

| type of patterns | patterns |
| :--- | :--- |
| Co | $\{a\}: 3,\{a\}\{b\}: 2,\{a\}\{b\}\{c\}: 2,\{b\}: 3,\{b\}\{c\}: 3,\{b\}\{c\}\{d\}: 2,\{c\}: 3$, <br> $\{c\}\{d\}: 2,\{d\}: 2$ |
| Co $_{1} R_{0,6}$ | $\{a\}: 3,\{a\}\{b\}: 3,\{a\}\{b\}\{c\}: 3,\{a\}\{b\}\{c\}\{d\}: 3,\{b\}: 3,\{b\}\{c\}: 3$, <br> $\{b\}\{c\}\{d\}: 2,\{c\}: 3,\{c\}\{d\}: 2,\{d\}: 2$ |
| ClCo | $\{a\}\{b\}\{c\}: 2,\{b\}\{c\}: 3,\{b\}\{c\}\{d\}: 2$ |
| $\mathrm{ClCo}_{1}$ | $\{a\}\{b\}\{c\}: 3,\{a\}\{b\}\{c\}\{d\}: 2,\{b\}\{b\}: 2$ |
| $\mathrm{ClO}_{1} R_{0,6}$ | $\{a\}\{b\}\{c\}: 3,\{a\}\{b\}\{c\}\{d\}: 2$ |
| $\mathrm{Cl}_{0,65} \mathrm{Co}_{1} R_{0,6}$ | $\{a\}\{b\}\{c\}\{d\}: 2$ |

Table 9：Datasets characteristics

| dataset | \＃seq． | \＃items | avg．len． | max．len． | std．dev．len． |
| :---: | ---: | ---: | ---: | ---: | ---: |
| BMS 1 Gazelle | 59,601 | 499 | 6.0 | 535 | 9.7 |
| LKosarak | 305,281 | 32,138 | 8.1 | 2,498 | 23.6 |
| Kosarak | 990,002 | 41,270 | 8.1 | 2,498 | 23.7 |
| Jobs 60 K | 60,000 | 92,394 | 77.0 | 1,667 | 56.8 |

Table 10: Memory usage on the three datasets w.r.t min_sup (MB)

| dataset | Minsupp(\%) | FI | CCSpan | CM-Clasp | $C 3_{0} R o$ | $C 3_{\infty} R o$ |
| :--- | :--- | :--- | ---: | ---: | ---: | ---: |
| BMS 1 Gazelle | 0.05 | 4 | 250 | 4,200 | $<50$ | $<50$ |
|  | 0.005 | 150 | 250 | 4,500 | 200 | 200 |
|  | 0.0005 | 374 | 300 | - | 200 | 250 |
| LKosarak | 0.05 | 10 | - | 5,000 | 500 | 500 |
|  | 0.005 | 158 | - | 6,700 | 500 | 500 |
|  | 0.0005 | 2303 | - | - | 550 | 550 |
| Kosarak | 0.05 | 10 | - | 10,600 | 1,200 | 1,200 |
|  | 0.005 | 156 | - | 10,600 | 1,200 | 1,200 |
|  | 0.0005 | 2297 | - | - | 1,300 | 1,300 |

Table 12: mined activities with or without C3Ro in Jobs 60 K

| activity id | configuration | activities | absolute support |
| :---: | :---: | :--- | ---: |
| $\# 1$ | without C3Ro | managing sales teams | 1712 |
| $\# 2$ | without C3Ro | managing sales teams effectively | 1352 |
| $\# 2$ | with C3Ro | managing sales teams effectively | 1536 |
| $\# 3$ | with C3Ro | managing sales teams distribution channels | 384 |


[^0]:    Email addresses: yacine. abboud@gmail.com (Y. Abboud) armelle.brun@loria.fr (A. Brun), anne.boyer@loria.fr (A. Boyer)

[^1]:    ${ }^{1}$ http://www.philippe-fournier-viger.com/spmf/index.php

