RAPPORT DE RECHERCHE

L R I

AN ALGORITHM FOR MINING MINIMAL SEQUENTIAL NUGGETS OF KNOWLEDGE

RANCE B / LISACEK F / FROIDEVAUX C

Unité Mixte de Recherche 8623
CNRS-Université Paris Sud – LRI

10/2007

Rapport de Recherche N° 1476

CNRS – Université de Paris Sud
Centre d'Orsay
LABORATOIRE DE RECHERCHE EN INFORMATIQUE
Bâtiment 490
91405 ORSAY Cedex (France)

An Algorithm for Mining Minimal Sequential Nuggets of Knowledge

Bastien Rance[1], Frédérique Lisacek[2],
Christine Froidevaux[1]
[1]LRI; Univ. Paris-Sud, CNRS UMR 8623;
F-91405 Orsay, France
{rance, chris}@lri.fr
[2]Proteome Informatics Group, Swiss Institute of Bioinformatics,
Geneva, Switzerland
frederique.lisacek@isb-sib.ch

September 19, 2007

Abstract

We present the notion of sequential association rule and introduce Sequential Nuggets of Knowledge as sequential association rules with possible low support and good quality, which may be highly relevant to scientific knowledge discovery. Then we propose the algorithm SNK that mines some interesting subset of sequential nuggets of knowledge. We have proved SNK to be both sound and complete with respect to that subset. A first implementation in Java is freely available on the web[1].

Nous présentons la notion de règle d'association séquentielle de connaissance et introduisons les pépites séquentielles de connaissance. Il s'agit d'une règle d'association avec un faible support et une très bonne qualité, qui peut être très pertinente pour la découverte de connaissances scientifiques. Nous proposons l'algorithme SNK qui recherche des sous ensembles intéressants de pépites séquentielles de connaissance, et prouvons qu'il est correct et complet pour ces sous-ensemble. Une première implémentation en Java est disponible sur notre site web[1].

1 Introduction

Mining the collection of records in a large database to find out association rules is a classical problem introduced by [1] that has received a great deal of attention. Association rules are expressions of the form $A \rightarrow B$, where A and B are disjoint itemsets. Frequent sequential patterns mining

[1] `http://www.lri.fr/~rance/SNK/`

was introduced in [2] in the case where the data stored in the database are relative to behavioural facts that occur over time as a refinement of frequent pattern mining that accommodates ordered items. It is an active research field in data mining that is applied in various domains including, among others, analysis of customer shopping sequences, web usage mining, medical processes, DNA sequences.

In this technical report, we introduce the notion of sequential association rule which is based on the notion of interestingness measure. Unlike common approaches, we are only interested in producing rules whose consequent belongs to some predefined set of items (target items), disjoint from the set of the items present in the antecedent. We want to detect tight associations between antecedents of rules and their consequent rather than rules with high support. Thus as in [14], we also search for significant rare data that co-occur in relatively high association with the specific data. Namely discovering close dependencies between facts that almost always co-occur is informative, even if these facts are not frequent in the database. In contrast, associations with large support cannot be surprising since they are relative to a large part of the objects ([3], [8]). Unexpected associations are interesting because they may reveal an aspect of the data that needs further study [7].

We determine the relevance of a rule merely by its value for some *interestingness measure*. We will consider several interestingness measures because not all measures are equally good at capturing the dependencies between the facts and no measure is better than others in all cases [12]. Then we introduce *Sequential Nuggets of Knowledge* as sequential association rules that may have a low support in the database but are highly relevant for some interestingness measure. Finally, not all Sequential Nuggets of Knowledge, but only the maximal ones are searched for. The rational is to reduce the number and the length of rules, assuming that such rules correspond in some way to a typical signature of the objects, that is, represent concise characteristics of the studied objects. Moreover they are easier to analyse for human experts.

Maximal Sequential Nuggets of Knowledge could be used for example to improve the organisation of a web site. Given the log (list of tuples <IP address, date, visited web page>) of visitors to our university web site, IP addresses could be used to identify different profiles of users: e.g. students of our university, researchers from other universities, visitors from the remainder of the world. If we could discover typical signatures for each profile, we would improve our web site organisation by adding hyperlinks between different pages and would simplify the navigation for the users.

In this technical report, we present the algorithm SNK which calculates the most general Sequential Nuggets of Knowledge. Sequential Nuggets of Knowledge express context-sensitive sequential constraints that are mostly verified in a sub-class of objects as opposed to another sub-class.

The remainder of the paper is organised as follows. In section 2 we introduce the fundamental concepts underlying the notion of Sequential

Nuggets of Knowledge. We present and study the algorithm SNK (section 3) that computes these nuggets. We report related work and conclude by discussing our results and giving some perspectives (section 4).

2 Basic concepts

2.1 Definitions

We aim at discovering dependencies between the descriptions of objects in terms of sequences of items in relation with some specific target item. We denote by IDT the set of identifiers of the objects and by T the set of the target items. Let I be the set of all *items* (boolean attributes). The sets I and T are supposed to be disjoint. An *itemset* is any subset of I.

The following notion of sequence is borrowed from [2]. A *sequence* s on I is an ordered list of itemsets, denoted by $\langle E_1, E_2, ..., E_l \rangle$, where $E_i \subseteq I, 1 \leq i \leq l$. Note that an itemset can have multiple occurrences in a sequence.

The *size* of a sequence s is the number of itemsets in s and is written $|s|$. A sequence $s = \langle E_1, E_2, ..., E_n \rangle$ is called a *subsequence* of another sequence $s' = \langle F_1, F_2, ..., F_m \rangle$, denoted $s \sqsubseteq s'$, if and only if there exist integers $j_1, ..., j_n$, such that $1 \leq j_1 < j_2 < ... < j_n \leq m$ and $E_1 \subseteq F_{j_1}$, $E_2 \subseteq F_{j_2}$, ... , $E_n \subseteq F_{j_n}$, where $\subseteq$ denotes the classical inclusion between sets. We will say that s' *contains* s. If s and s' are distinct sequences such that $s \sqsubseteq s'$, we will write $s \sqsubset s'$.

Let $s = \langle E_1, E_2, ..., E_n \rangle$ and $s' = \langle F_1, F_2, ..., F_m \rangle$ be two sequences on I. We will denote by $s \cdot s'$ the sequence resulting from the concatenation of the two sequences: $s \cdot s' = \langle E_1, E_2, ..., E_n, F_1, F_2, ..., F_m \rangle$.

We define a *categorised sequence database* as a set CSD of tuples $\langle sid, s, tg \rangle$, $sid \in IDT$, $tg \in T$, where sid is the object identifier, s the sequence of itemsets from I describing it and tg the target item associated to it. A tuple $\langle sid, s, tg \rangle$ is said to *contain* a sequence s' if and only if s' is a subsequence of s.

Running example:

$CSD =$

	id	seq	target
$\{\alpha_1 =$	$\langle id_1,$	$\langle a, b, f, c, e, f, g \rangle$	$, tg_1 \rangle,$
$\alpha_2 =$	$\langle id_2,$	$\langle a, e, b, h, c, f, g \rangle$	$, tg_1 \rangle,$
$\alpha_3 =$	$\langle id_3,$	$\langle c, e, a, b, e, g, f \rangle$	$, tg_2 \rangle,$
$\alpha_4 =$	$\langle id_4,$	$\langle c, e, a, b, e, g, f, a, e, b, f, d \rangle$	$, tg_2 \rangle\}$

In CSD the sequence $\langle b, e, f \rangle$ is a subsequence of $\langle a, b, f, c, e, f, g \rangle$ and α_1 contains the sequence $\langle b, e, f \rangle$. In this example all the itemsets are singletons denoted by their unique element, which is not required in the general definition.

We introduce the notion of sequential association rule as a combination of classical association rules and sequential patterns. Formally, a *sequential association rule* r on CSD is an implication of the form $ANT \rightarrow CONS$, where ANT is a sequence of itemsets from I and $CONS$

an element of T. We call ANT (resp. $CONS$) the *antecedent* (resp. *consequent*) of r and write $ant(r)$ (resp. $cons(r)$).

The *support* of a sequential association rule r in a database CSD is defined as the number of tuples of CSD that contain both its antecedent and its consequent. Formally we have: $support_{CSD}(ANT \rightarrow CONS) = |\{\langle sid, s, tg\rangle \in CSD \ s.t. \ (ANT \sqsubseteq s) \wedge (CONS = tg)\}|$.
Note that the items in ANT need not be consecutive in s, in order to be supported by the tuple.

Example: $support_{CSD}(\langle a, b, f\rangle \rightarrow tg_1) = 2$

The *confidence* of a sequential association rule r in the database CSD indicates amongst all the tuples of CSD containing its antecedent the fraction in which its consequent appears. $conf_{CSD}(ANT \rightarrow CONS) =$

$$\frac{|\{\langle sid, s, tg\rangle \in CSD \ s.t. \ (ANT \sqsubseteq s) \wedge (CONS = tg)\}|}{|\{\langle sid, s, tg\rangle \in CSD \ s.t. \ ANT \sqsubseteq s\}|}$$

Example: $conf_{CSD}(\langle a, b, f\rangle \rightarrow tg_1) = 0.5$; $conf_{CSD}(\langle a, b, f, g\rangle \rightarrow tg_1) = 1$.

A sequential association rule r_1 is said to *contain* another rule r_2, written $(r_2 \preceq r_1)$, if and only if $cons(r_1) = cons(r_2)$ and $ant(r_2) \sqsubseteq ant(r_1)$. We also say that r_2 is *more general* than r_1. If $r_1 \neq r_2$ and $r_2 \preceq r_1$ we will write $r_2 \prec r_1$.

We now focus on the main notion of this paper, namely *Sequential Nuggets of Knowledge*. We introduce them as sequential association rules with possible low support but with hight quality. Minimal support is required in order not to discover strong associations that involve only a few objects, which may come from noise.

A *sequential nugget of knowledge* is defined as a sequential association rule r in CSD such that its support is no less than some threshold and its quality is no less than to some other threshold.

In the applications we have foreseen, objects are merely described by sequences of items, so that sequences of itemsets are unnecessarily complicated. Therefore, in the remainder of the paper, we will consider only sequences where itemsets have a single item. The definition of subsequence can be rewritten in a simpler form where inclusion is replaced by equality.

2.2 Interestingness measures

Identifying sequences of variables that are strongly correlated and building relevant rules with those variables is a challenging task. Interestingness measures help to estimate the importance of a rule: they can be used for pruning low utility rules, or ranking and selecting interesting rules. Selecting a good measure allows to reduce time and space costs during the mining process ([12], [7]). As pointed earlier, all the interestingness measures do not capture the same kind of association. For example, using a support-confidence approach, a rule $ANT \rightarrow CONS$ may may be considered as important, even if $CONS$ is often found without ANT. The

distribution of examples of $\overline{ANT}$ between $CONS$ and $\overline{CONS}$ is not taken into account. In our work we mainly studied, besides confidence, another measure which is well adapted to our data, Zhang's measure as it takes into consideration the counter-examples [16].

[8] and [7] suggest a number of key properties to be examined for selecting the right measure that best suits the data. Note that while support satisfies anti-monotonicity (if $r \preceq r'$ then $support_{CSD}(r') \leq support_{CSD}(r)$), not all interestingness measures satisfy monotonicity (if a rule is considered to be relevant any of its specialisations is relevant too).

2.3 Postfix-projection

The method proposed for mining sequential nuggets of knowledge follows the approach of [11] for sequential patterns. We recursively project the initial categorised sequential database into a set of smaller categorised sequential databases, thus generating projected databases by growing prefixes.

Let CSD be a categorised sequential database, $\alpha = \langle sid_1, \langle e_1 ... e_n \rangle, c_1 \rangle$ a tuple of CSD and $s' = \langle e'_1 ... e'_m \rangle$ a sequence with $m \leq n$. s' is called a *prefix* of α if and only if $\forall i, 1 \leq i \leq m$, $e'_i = e_i$.

Example (continued): The sequence $\langle a, b, f \rangle$ is a prefix of α_1.

Let $\alpha = \langle sid, s, tg \rangle$ be a tuple of CSD. We denote id, seq and $target$ the methods which return respectively the identifier, the sequence and the target of α: $id(\alpha) = sid$, $seq(\alpha) = s$ and $target(\alpha) = tg$.

The notion of s'-projection corresponds to the longest subsequence having s' as a prefix. Let α be a tuple and s' be a sequence such that $s' \sqsubseteq seq(\alpha)$. A tuple $\alpha' = \langle id(\alpha'), seq(\alpha'), target(\alpha') \rangle$ is the *s'-projection* of α if and only if (1) $id(\alpha') = id(\alpha)$, (2) $seq(\alpha') \sqsubseteq seq(\alpha)$, (3) $target(\alpha') = target(\alpha)$, (4) s' is *a prefix* of α' and (5) $\nexists \alpha''$ a tuple s.t. $seq(\alpha') \sqsubset seq(\alpha'')$ and $seq(\alpha'') \sqsubseteq seq(\alpha)$ and s' is a prefix of α''.

Note that with such a definition only the subsequence of $seq(\alpha)$ prefixed with the first occurrence of s' should be considered for α'.

Example (continued):

$\langle id_1, \langle a, b, f, c, e, f, g \rangle, tg_1 \rangle$ is an abf-projection of α_1, while $\langle id_1, \langle a, b, f, g \rangle, tg_1 \rangle$ is not because (5) is not satisfied. Similarly, $\langle id_4, \langle a, b, f, a, e, b, f, d \rangle, tg_2 \rangle$ is an abf-projection of α_4, while $\langle id_4, \langle a, b, f, d \rangle, tg_2 \rangle$ is not because of (5).

The s'-projection of α, if it exists (i.e. if s' can be a prefix of a tuple whose sequence is contained in α) is unique. It is *the* s'-projection of α.

Let α be a tuple of CSD and let $s = \langle e_1, ..., e_n \rangle$ be a sequence on I. Let $\alpha' = \langle id_1, \langle e_1, ..., e_n, e_{n+1}, ..., e_{n+p} \rangle, tg_1 \rangle$ be the s-projection of α, where s is a prefix of α'. Then $\gamma = \langle id_1, \langle e_{n+1}, ..., e_{n+p} \rangle, tg_1 \rangle$ is the *s-postfix* of α'. If $p > 0$, then the s-postfix has a sequence of size > 0: it is said to be not empty and is denoted by α/s. Note that γ satisfies: $seq(\alpha') = s \cdot seq(\gamma)$.

The *s-projected database*, denoted by $s - postfix(CSD)$, is defined as follows:

$s-$postfix$(CSD) = \{(\alpha/s),\ \alpha \in CSD\}$

Running example :

$abf-\text{postfix}(CSD) =$

id	seq	target
$\{\langle id_1,$	$\langle c, e, f, g\rangle$	$, tg_1\rangle,$
$\langle id_2,$	$\langle g\rangle$	$, tg_1\rangle,$
$\langle id_4,$	$\langle a, e, b, f, d\rangle$	$, tg_2\rangle\}$

The recursive principle of our algorithm is based on the following property:

Property 1:

Let CSD be a categorised database. Let s_1 and s_2 be any sequences on I, and let r be any sequential association rule. Then:

(i) $s_2-\text{postfix}(s_1-\text{postfix}(CSD)) = s_1 \cdot s_2-\text{postfix}(CSD)$

(ii) $\text{support}_{s_1 \cdot s_2-\text{postfix}(CSD)}(r) = \text{support}_{CSD}((s_1 \cdot s_2 \cdot ant(r)) \rightarrow cons(r))$

(iii) $\text{support}_{CSD}(r) \geq \text{support}_{s_1-\text{postfix}(CSD)}(r)$.

Proof (1):

(i) $s_2-\text{postfix}(s_1-\text{postfix}(CSD)) = s_1 \cdot s_2-\text{postfix}(CSD)$

I)

Let $\gamma_2 \in s_2 - \text{postfix}(s_1 - \text{postfix}(CSD))$. Let us show that $\gamma_2 \in s_1 \cdot s_2 - \text{postfix}(CSD)$.

If $\gamma_2 \in s_2 - \text{postfix}(s_1 - \text{postfix}(CSD))$ then by definition :

s_2 is a sequence

there exists a tuple $\alpha_2 \in s_1 - \text{postfix}(CSD)$

and there exists a tuple α_2' such that α_2' is the $s_2 -$ projection of α_2

But,

$\alpha_2' = s_2 - \text{projection}(\alpha_2)$ means

1) s_2 is a prefix of α_2', more precisely, $\text{seq}(\alpha_2') = s_2 \cdot \text{seq}(\gamma_2)$ $(\Diamond)$

2) $\text{id}(\alpha_2) = \text{id}(\alpha_2') = \text{id}(\gamma_2)$ (∇)

3) $\text{target}(\alpha_2) = \text{target}(\alpha_2') = \text{target}(\gamma_2)$ $(\heartsuit)$

4) $\text{seq}(\alpha_2') \sqsubseteq \text{seq}(\alpha_2)$ $(\Box)$

and there exists no tuple α_2'' such that $\text{seq}(\alpha_2') \sqsubset \text{seq}(\alpha_2'') \sqsubseteq \text{seq}(\alpha_2)$ and s_2 is a prefix of α''.

Now $\alpha_2 \in s_1 - \text{postfix}(CSD)$ means that

s_1 is a sequence

there exists a tuple $\alpha_1 \in CSD$

and there exists α_1' a tuple such that α_1' is the $s_1 -$ projection of α_1

But,

$\alpha_1' = s_1 - \text{projection}(\alpha_1)$ means :

1) s_1 is a prefix of α_1', more precisely, $\text{seq}(\alpha_1') = s_1 \cdot \text{seq}(\alpha_2)$ $(\Diamond\Diamond)$

2)$\text{id}(\alpha_1) = \text{id}(\alpha_1') = \text{id}(\alpha_2)$ $(\nabla\nabla)$

3) $\text{target}(\alpha_1) = \text{target}(\alpha_1') = \text{target}(\alpha_2)$ $(\heartsuit\heartsuit)$

4) $\text{seq}(\alpha_1') \sqsubseteq \text{seq}(\alpha_1)$ $(\Box\Box)$

and there exists no tuple α_1'' such that $\text{seq}(\alpha_1') \sqsubset \text{seq}(\alpha_1'') \sqsubseteq \text{seq}(\alpha_1)$ and s_1 is a prefix of α_1''.

Let us consider $\alpha_1 \in CSD$, and let us show that $\gamma_2 = \alpha_1/s_1 \cdot s_2$.

Consider $\delta' = \langle id(\gamma_2), \langle s_1 \cdot s_2 \cdot seq(\gamma_2)\rangle, target(\gamma_2)\rangle$.

Let us show that δ' is the $s_1 \cdot s_2-$projection of α_1

(1) $\text{id}(\alpha_1) = \text{id}(\alpha_2)(\nabla\nabla)$ and $\text{id}(\alpha_2) = \text{id}(\gamma_2)(\nabla)$. Therefore $\text{id}(\delta') =$

$\mathrm{id}(\alpha_1)$
(2) $s_2 \cdot \mathrm{seq}(\gamma_2) \sqsubseteq \mathrm{seq}(\alpha_2)(\Diamond)$ and $(\Box)$
and $s_1 \cdot \mathrm{seq}(\alpha_2) \sqsubseteq \mathrm{seq}(\alpha_1)(\Diamond\Diamond)$ and $(\Box\Box)$
Therefore, $s_1 \cdot s_2 \cdot \mathrm{seq}(\gamma_2) \sqsubseteq \mathrm{seq}(\alpha_1)$, i.e., $\mathrm{seq}(\delta') \sqsubseteq \mathrm{seq}(\alpha_1)$
(3) $\mathrm{target}(\alpha_1) = \mathrm{target}(\alpha_2)(\heartsuit\heartsuit)$ and $\mathrm{target}(\alpha_2) = \mathrm{target}(\gamma_2)(\heartsuit)$. Therefore $\mathrm{target}(\delta') = \mathrm{target}(\alpha_1)$
(4) $s_1 \cdot s_2$ is clearly a prefix of δ'.
(5) We reason ad absurdum. Assume that there exists a tuple δ'' such that
$\mathrm{seq}(\delta'') = s_1 \cdot s_2 \cdot \mathrm{seq}(\varepsilon) \sqsubseteq \mathrm{seq}(\alpha_1)$
Assume that $\mathrm{seq}(\gamma_2) \not\sqsubseteq \varepsilon$.
Necessarily there must be in α_1 another occurrence of s_2 on the left of that of δ'. But as α_2' is the $s_2 - \mathrm{projection}(\alpha_2)$, that left occurrence of s_2 must begin in α_1 before the beginning of α_2 (property (5) of the s_2-projection). Consequently there must be in α_1 another occurrence of s_1 on the left of that of δ'. A contradiction with the fact that α_2 is the $s_1 -$ projection of α_1. We can conclude that there can be no such δ''.

Thus $\delta' = s_1 \cdot s_2 - \mathit{projection}(\gamma_2)$ and $\gamma_2 = \alpha_1/s_1 \cdot s_2$. As $\alpha_1 \in CSD, \gamma_2 \in s_1 \cdot s_2 \cdot \mathrm{postfix}(CSD)$. $(\Box)$

II)
Let $\gamma_4 \in s_1 \cdot s_2 - \mathrm{postfix}(CSD)$. Let us show that $\gamma_4 \in s_2 - \mathrm{postfix}(s_1 - \mathrm{postfix}(CSD))$.

If $\gamma_4 \in s_1 \cdot s_2 - \mathrm{postfix}(CSD)$ by definition there exists $\alpha_4 \in CSD$ such that $\gamma_4 = \alpha_4/s_1 \cdot s_2$.

Let $\beta = \alpha_4/s_1$ then clearly $\beta \in s_1 - \mathrm{postfix}(CSD)$.
Let $\delta = \beta/s_2$ then clearly $\delta \in s_2 - \mathrm{postfix}(s_1 - \mathrm{postfix}(CSD))$.
Let us show that $\delta = \alpha_4/s_1 \cdot s_2$.
We have $\mathrm{id}(\delta) = \mathrm{id}(\alpha_4)$ and $\mathrm{target}(\delta) = \mathrm{target}(\alpha_4)$.

Let $\delta' = \langle \mathrm{id}(\alpha_4), \langle s_1 \cdot s_2 \cdot \mathrm{seq}(\delta)\rangle, \mathrm{target}(\alpha_4)\rangle$
$\mathrm{seq}(\delta') = s_1 \cdot s_2 \cdot \mathrm{seq}(\delta)$
Obviously δ is the $s_1 \cdot s_2-$postfix of δ'
Let us show that δ' is the $s_1 \cdot s_2-$projection of α_4 :
(1) $\mathrm{id}(\delta') = \mathrm{id}(\alpha_4)$
(2) Is $\mathrm{seq}(\delta') \sqsubseteq \mathrm{seq}(\alpha_4)$?
$\beta = \alpha_4/s_1$ implies that $s_1 \cdot \mathrm{seq}(\beta) \sqsubseteq \mathrm{seq}(\alpha_4)$
$\delta = \beta/s_2$ implies that $s_2 \cdot \mathrm{seq}(\delta) \sqsubseteq \mathrm{seq}(\beta)$
Therefore $s_1 \cdot s_2 \cdot \mathrm{seq}(\delta) \sqsubseteq \mathrm{seq}(\alpha_4)$ and thus
$\mathrm{seq}(\delta') \sqsubseteq \mathrm{seq}(\alpha_4)$
(3) $\mathrm{target}(\delta') = \mathrm{target}(\alpha_4)$
(4) $s_1 \cdot s_2$ is clearly a prefix of δ'
(5) Let us show that there exists no tuple δ'' s.t. $\mathrm{seq}(\delta'')$ has $s_1 \cdot s_2$ as a prefix and $\mathrm{seq}(\delta') \sqsubset \mathrm{seq}(\delta'') \sqsubseteq \mathrm{seq}(\alpha_4)$.
Now since $\beta = \alpha_4/s_1$, β is the longuest subsequence of α_4 after s_1 and since $\delta = \beta/s_2$, δ is the longuest subsequence of β after s_2. Therefore δ is the longuest subsequence of α_4 after $s_1 \cdot s_2$. Consequently there can be

no δ'' such that $seq(\delta') \not\sqsubset \text{seq}(\delta') \sqsubseteq \text{seq}(\alpha_4)$.
We can conclude that $\delta = \alpha_4/s_1 \cdot s_2 = \gamma_4$ and therefore that $\gamma_4 \in s_2 - \text{postfix}(s_1 - \text{postfix}(CSD))$.

Proof (2)

Let us show that $\text{support}_{s_1 \cdot s_2 - \text{postfix}(CSD)}(r) = \text{support}_{CSD}((s_1 \cdot s_2 \cdot \text{ant}(r)) \rightarrow \text{cons}(r))$.

(A) First we prove that if $\gamma = \langle id, s, tg \rangle \in s_1 \cdot s_2 - \text{postfix}(CSD)$ is such that $\text{ant}(r) \sqsubseteq s \wedge \text{cons}(r) = tg$ then there exists $\alpha \in CSD$ s.t. $s_1 \cdot s_2 \cdot \text{ant}(r) \sqsubseteq \text{seq}(\alpha)$ and $\text{cons}(r) = tg$.
If $\gamma \in s_1 \cdot s_2 - \text{postfix}(CSD)$ then there exists $\alpha \in CSD$ s.t. $\text{target}(\gamma) = \text{target}(\alpha)$ and $s_1 \cdot s_2 \cdot \text{seq}(\gamma) \sqsubseteq \text{seq}(\alpha)$.
Therefore $\text{target}(\alpha) = \text{cons}(r) = tg$.
As $\text{ant}(r) \sqsubseteq \text{seq}(\gamma)$, $s_1 \cdot s_2 \cdot \text{ant}(r) \sqsubseteq s_1 \cdot s_2 \cdot \text{seq}(\gamma)$ et $s_1 \cdot s_2 \cdot \text{ant}(r) \sqsubseteq \text{seq}(\alpha)$.
As a result, $\alpha \in CSD$ and α supports the rule $s_1 \cdot s_2 \cdot \text{ant}(r)$.

(B) Now let $\alpha = \langle id, s, tg \rangle \in CSD$ s.t. $s_1 \cdot s_2 \cdot \text{ant}(r) \sqsubseteq s$ and $\text{cons}(r) = tg$.
Let $\gamma = s_1 \cdot s_2 - \text{postfix}(\alpha)$, let us show that γ supports r, i.e. $\text{ant}(r) \sqsubseteq \text{seq}(\gamma)$
(1) $\text{target}(\gamma) = \text{target}(\alpha) = tg = \text{cons}(r)$
(3) $\gamma = s_1 \cdot s_2 - \text{postfix}(\alpha)$ means that there exists $\alpha' \in s_1 \cdot s_2 - \text{projection}(\alpha)$ such that:
$\text{seq}(\alpha') = s_1 \cdot s_2 - \text{seq}(\gamma)$
$\text{seq}(\alpha') \sqsubseteq \text{seq}(\alpha)$
and $\nexists \alpha''$ a tuple s.t. $\text{seq}(\alpha') \sqsubset \text{seq}(\alpha'')$ and $\text{seq}(\alpha'') \sqsubseteq \text{seq}(\alpha)$ and $s_1 \cdot s_2$ is a prefix of α''.
We have $\text{seq}(\alpha') \sqsubseteq \text{seq}(\alpha) = s$, i.e., $s_1 \cdot s_2 \cdot \text{seq}(\gamma) \sqsubseteq s$.
But $s_1 \cdot s_2 \cdot \text{ant}(r) \sqsubseteq s$ by hypothesis.
As $\text{seq}(\gamma)$ is the longest subsequence of $\text{seq}(\alpha)$ having $s_1 \cdot s_2$ as a prefix, $s_1 \cdot s_2 \cdot \text{ant}(r) \sqsubseteq \text{seq}(\alpha)$ implies that $\text{ant}(r) \sqsubseteq \text{seq}(\gamma)$. Therefore γ supports r.

In conclusion, for each $\gamma \in s_1 \cdot s_2 - \text{postfix}(CSD)$ s.t. γ supports r, there exists $\alpha \in CSD$ that supports $s_1 \cdot s_2 \cdot ant(r) \rightarrow \text{cons}(r)$ and conversely.
Therefore the property (ii) is established.

Proof (3)
(iii) $\text{support}_{CSD}(r) \geq \text{support}_{s_1 - \text{postfix}(CSD)}(r)$.
Let us show that for all $\gamma \in s_1 - \text{postfix}(CSD)$ that supports r, there exists $\alpha \in CSD$ that supports r.

$\gamma = \langle id, s, tg \rangle \in s_1 - \text{postfix}(CSD)$ is such that $\text{ant}(r) \sqsubseteq s \wedge \text{cons}(r) = tg$. Then there exists $\alpha \in CSD$ such that $\text{target}(\gamma) = \text{target}(\alpha)$ and $s_1 \cdot \text{seq}(\gamma) \sqsubseteq \text{seq}(\alpha)$. As a result, $\alpha \in CSD$ and α supports the rule $(s_1.\text{ant}(r) \rightarrow \text{cons}(r))$.

In conclusion, for each $\gamma \in s_1 - \text{postfix}(CSD)$ s.t. γ supports r, there exists $\alpha \in CSD$ that supports $s_1 \cdot \text{ant}(r) \rightarrow \text{cons}(r)$.
Therefore the property (iii) is established.

3 SNK Algorithm

3.1 Specification and pseudo-code

Now we present SNK, an algorithm which mines the most general sequential nuggets of knowledge from a categorised sequential database, given some thresholds specified by the user.

SNK method
Parameters:
In: CSD a categorised sequential database; min_supp a support threshold; IM an interestingness measure; min_meas an IM value threshold;
Out: $RESULTS$ the set of the most general Sequential Nuggets of Knowledge;
Method used: SNKrec;
Begin
$RESULTS = \emptyset$; ST = the set of all target items of T present in CSD;
Foreach y in ST **do**
//sequential nuggets of knowledge targeted on y are searched for
S_y = the set of all tuples of CSD having y as a target;
SNKrec(S_y,y,min_supp,IM,min_meas,$\langle\rangle$,$RESULTS$) **endfor end_SNK**;

SNKrec method
// generates rules r of the form $(p \cdot x) \rightarrow y$, where x is any item occurring in S and p the prefix used; updates $RESULTS$ with r in order to get only the most general sequential nuggets of knowledge; calls recursively itself on the x-projected database of S if r has good support but bad interestingness measure value
Parameters:
In: S a set of tuples having y as a target; min_supp, IM, min_meas;
p the sequence used as a prefix;
In/Out: $RESULTS$ a set of Sequential Nuggets of Knowledge s.t. / $\exists r_1, r_2 \in RESULTS$ with $r_1 \prec r_2$;
Methods used:
add_rule; //add_rule(r,RES) adds rule r to RES unless if r is less general than or equal to some rule in RES and removes from RES any rule that is less general than r.
support; // $\text{support}_S(r)$ evaluates the support of r in S. measure; // $\text{measure}_{IM,CSD}$(r) evaluates the value of r for IM in CSD
Begin SI = the set of all items of I occurring in elements of S;
Foreach x in SI **do**
if $\text{support}_S(x \rightarrow y) \geq min\_supp$ **then**
if $\text{measure}_{IM,CSD}((p \cdot x) \rightarrow y) \geq min\_meas$ **then**
$RESULTS$ = add_rule($(p \cdot x) \rightarrow y$,$RESULTS$)
else if x-postfix(S) $\neq \emptyset$ **then**

SNKrec(x-postfix(S),y,min_supp,IM,min_meas,$p{\cdot}x$,$RESULTS$)
endifendifendifendifendfor end_SNKrec;

add_rule method
//add_rule(r,RES) adds rule r to RES if r is not more general or equal to any rule in RES and removes from RES any rule that is more general than r
Parameters:
In: r a rule;
In/Out: RES a set of rules s.t. $\nexists \rho_1, \rho_2 \in RES$ with $\rho_1 \prec \rho_2$;
Begin max_found = false; $i = 1$; let RES be $\{r_1, ..., r_n\}$;
while($i \leq n$ **and** max_found = false) **do**
if($r \prec r_i$) **then** $RES = RES \setminus \{r_i\}$
else if($r_i \preceq r$) **then** max_found = true **endifendif**;
$i = i + 1$
endwhile;
if max_found = false **then** $RES = RES \cup \{r\}$ **endif end_add_rule**;

3.1.1 Complexity, completeness, soundness

Time complexity
The time complexity of SNK is related to the number of target items y of T present in CSD, and for each y in ST, to the number of recursive calls of SNKrec. Therefore we measure the complexity by estimating the number of tests (**if** x-postfix(S) $\neq \emptyset$) performed by SNKrec for some given y. The worst case for SNKrec occurs when all the rules generated have good support but bad measure, leading to a maximal number of recursive calls.
Let us consider the tree of the recursive calls of SNKrec and let rs_y be the depth of this tree: it is the length of the longest sequence of a tuple of S. Let $rs_{y,i}$ be the length of the longest sequence of a tuple of the projected database S at the i-th level. Clearly $rs_{y,i} \leq rs_y - i$. Let l_i be the maximal cardinal of the set SI considered at the i-th level. Then an upper bound for the number of tests at the i-th level is $\prod_{j=0}^{i-1} l_j$. Let $\sigma_{y,i}$ be the maximal number of tuples that can be found at the i-th level in any projected database S. Then building $x - postfix(S)$ at the i-th level requires $\mathcal{O}(\sigma_{y,i} \times rs_{y,i})$ operations. Finally an upper bound for the total number of operations performed by SNK to build projected databases is:

$$\mathcal{O}(\sum_{y \in ST} \sum_{i=1}^{rs_y} (\prod_{j=0}^{i-1} l_j) \text{Max}((\sigma_{y,i} \times rs_{y,i}), (|CSD| \times C_{rs_{y,0}}^{\text{int}(sr_{y,0}/2)})))$$

To each of these tests corresponds a calculation of support$_S((p{\cdot}x) \rightarrow y)$ which is no less than $\sigma_{y,i} \times rs_{y,i}$. The calculation of measure$_{IM,CSD}((p \cdot x) \rightarrow y)$ is straightforward: it results from a combination of a series of probabilities that are calculated once before SNK execution. At each recursive call of the algorithm, a categorised sequential database may product at most all the combinaison of subsequence of the longest sequence times the number of sequence rules. At each recursive call the cost is at

most either the cost of a postfix-projection or the cost of the add_rule method. With our depth-first search approach not all the projected databases need to be stored in memory unlike in a breadth-first search approach. Moreover the calculation of the different projected databases might be performed independently.
This analysis shows that the theoretical time complexity is very high in the worst case. However, in practice, for the applications foreseen, the SNK algorithm remains efficient because the size of the projected databases decreases very quickly.

Soundness and completeness

Theorem 1 ***(Soundness)***
Let CSD be a categorised sequential database of records. Let $RESULTS$ be the set of all sequential nuggets of knowledge returned by SNK($CSD, min\_supp, IM, min\_meas$) for some interestingness measure IM. Then:
(1) The support value in CSD of each $r \in RESULTS$ is no less than min_supp and the corresponding interestingness measure value in CSD is no less than min_meas.
(2) For any $r \in RESULTS$ there exists no rule r' on CSD s.t. $supp_{CSD}(r') \geq min\_supp$, $meas_{IM,CSD}(r') \geq min\_meas$ and $r' \prec r$.

Proof:
In the SNKrec method add_rule is called for rule $r = (p \cdot x) \rightarrow y$ if $\text{support}_s(x \rightarrow y) \geq min\_supp$ (a) and
$\text{measure}_{IM,CSD}(p \cdot x \rightarrow y) \geq min\_meas$ (b)

(1) Let r be a rule in $RESULTS$. It was added to $RESULTS$ (and not deleted) by the add_rule method called by SNKrec for somes values of the parameters S and p. Let k be the number of the recursive calls needed.
If $k = 1$ then r is of the form $x \rightarrow y$ and there is only one run of the SNKrec (the method being itself called from the main method SNK). Therefore p is the empty prefix and $S = S_y$, the set of tuples of CSD having y as a target.
If $k > 1$, then r is of the form $(x_1, x_2, ..., x_{k-1}, x_k) \rightarrow y$, since each call adds an item attribute in the antecedent of the rule.

The successive recursive calls of SNKrec have been performed with the following successive values of S: $S_1, S_2, ..., S_k$ with $S_1 = S_y$, $S_2 = x_1 - \text{postfix}(S_1)$, $S_3 = x_2 - \text{postfix}(S_2)$, ..., $S_k = x_{k-1} - \text{postfix}(S_{k-1})$.
Therefore $S_k = x_{k-1} - \text{postfix}(x_{k-2} - \text{postfix}(...(x_1 - \text{postfix}(S_y))...))$.
By property 1 (i), $S_k = x_1 \cdot x_2 \cdot ... \cdot x_{k-1} - \text{postfix}(S_y)$
Now by property 1 (ii),
$\text{support}_{x_1 \cdot x_2 \cdot ... \cdot x_{k-1} - \text{postfix}(S_y)}(x_k \rightarrow y) = \text{support}_{S_y}((x_1 \cdot ... \cdot x_{k-1}) \cdot x_k \rightarrow y)$
Since only the tuples of CSD having y as a target are useful for the calculation of the support of rules having y as a consequent, we get:

$$\text{support}_{x_1 \cdot x_2 \cdot \ldots \cdot x_{k-1} - \text{postfix}(S_y)}(x_k \rightarrow y) = \text{support}_{CSD}(r)$$

But since at the k^{th} call of SNKrec, rule r is handled by add_rule.

We have: $\text{support}_{S_{k-1}}(x_k \rightarrow y) \geq min\_supp$.
Therefore:

$$\text{support}_{CSD}(r) \geq min\_supp$$

(2) We reason ad absurdum. Assume that there exists a rule $r' \prec r$, with $r' = x_{p_1} \cdot \ldots \cdot x_{p_q} \rightarrow y \in CSD$, $p_1 < \ldots < p_q \in \{1, \ldots, k\}$ and $q < k$, such that $\text{support}_{CSD}(r') \geq min\_supp$ and $\text{measure}_{IM,CSD}(r') \geq min\_meas$.
There exists a succession of recursive calls of SNKrec that build the beginning of ant(r').

Assume that r' is completely built: then at the q^{th} call of SNKrec, add_rule would have been called and would have eliminated rule r since $r' \prec r$. It is impossible. Therefore, the process of building r' has been stopped at the j^{th} recursive call of SNKrec.
Let $r'' = x_{p_1} \cdot x_{p_2} \cdot \ldots \cdot x_{p_j} \rightarrow y$, with $j < q$.
In the same way as in (1), we can prove that $\text{support}_{CSD}(r'' = \text{support}_{S_j}(x_{p_j} \rightarrow y)$ where S_j is the value of the parameter S at the j^{th} call.
But since $r'' \prec r' \prec r$, we have $\text{support}_{CSD}(r'') \geq \text{support}_{CSD}(r) \geq min\_supp$ by definition of support.

Now, since no more recursive call is done, $\text{measure}_{IM,CSD}(r'') \geq min\_meas$.
Then SNKrec would have called the add_rule method which would have added r'' to $RESULTS$. As $r'' \prec r$, r would have been removed from $RESULTS$, a contradiction.
In conclusion, there exists no such rule r'.

Theorem 2 ***(Completeness)***
Let CSD be a categorised sequential database, IM an interestingness measure, min_supp a support threshold and min_meas an interestingness measure threshold for IM. Let $Q_{CSD,min\_supp,IM,min\_meas} = \{$ rules r on CSD satisfying $supp_{CSD}(r) \geq min\_supp$, $meas_{IM,CSD}(r) \geq min\_meas$ and s.t. $\nexists\, r'$ on CSD with $supp_{CSD}(r') \geq min\_supp$, $meas_{IM,CSD}(r') \geq min\_meas$ and $r' \prec r\}$. Then any rule of $Q_{CSD,min\_supp,IM,min\_meas}$ can be obtained by SNK($CSD, min\_supp, IM, min\_meas$).

Proof:

Let $r \in Q_{CSD,min\_supp,IM,min\_meas}$. Let us show that r can be obtained by SNK. Assume that r is of the form $x_1 \cdot \ldots \cdot x_k \rightarrow y$.

We show that r is returned by SNK after k recursive calls of SNKrec.
Base case : $k = 1$. Then SNK calls SNKrec with $S = S_y$ and $p = \emptyset$.
As $r \in Q$, $\text{support}_{CSD}(x \rightarrow y) \geq min\_supp$. But as already shown $\text{support}_{CSD}(x \rightarrow y) = \text{support}_{S_y}(x \rightarrow y)$. Since $r \in Q$, $\text{measure}_{IM,CSD}(x \rightarrow$

$y) \geq min\_meas$.
Therefore at the first call of SNKrec the add_rule method is called. Since $r \in Q$, there is no rule $r' \in RESULTS$ that can be contained by r. Consequently r is added to RESULTS and will stay in it until the algorithm is completed.

General case : $k > 1$. Let us show that SNKrec performs k consecutive recursive calls.
Let $S_1 = S_y, S_2 = x_1 - \text{postfix}(S_1), ..., S_k = x_{k-1} - \text{postfix}(S_{k-1})$ be the succesive values of parameter S in the k consecutive calls of SNKrec.
Let $r'_j = x_1 \cdot ... \cdot x_j \rightarrow y$ for $1 \leq j \leq k-1$
(a) Since $\text{support}_{CSD}(r) \geq min\_supp$, $\text{support}_{S_y} \geq min\_supp$.
By property 1 (i) and (ii):
$\text{support}_{S_j}(x_j \rightarrow y) = \text{support}_{S_y}(x_1 \cdot ... \cdot x_j \rightarrow y)$, $1 \leq j \leq k-1$
But $\text{support}_{S_y}(x_1 \cdot ... \cdot x_j \rightarrow y) \geq \text{support}_{S_y}(x_1 \cdot ... \cdot x_k \rightarrow y) \geq min\_supp$
Thus we get:
$\text{support}_{CSD}(r'_j) = \text{support}_{S_y}(r'_j) \geq min\_supp$.
(b) $r'_j \prec r$ for $1 \leq j \leq k-1$
(c) r'_j is a rule defined on CSD.
From (a), (b), (c) and the fact that $r \in Q$, we can conclude that $\text{measure}_{IM,CSD}(r'_j) < min\_meas$. Therefore, SNKrec will be called once more with parameter S_{j+1}.

We have shown that k recursive calls of SNKrec will be done, leading to rule r at the k^{th} call.
At that step, SNKrec will call add_rule method that will check whether r can be added to $RESULTS$. As $r \in Q$, there can be no rule $r' \in RESULTS$ such that $r' \prec r$, since any rule in $RESULTS$ satisfies support and interestingness measure requirements (cf. soundness part).

In conclusion r will be added to $RESULTS$, without possibility of removing it, and r will be returned by the main SNK method.

Theorems 1 and 2 give the following characterization of SNK output:

Theorem 3 ***(Soundness and Completeness of SNK)*** Let CSD be a categorised sequential database and IM be an interestingness measure. Given support and interestingness measure thresholds, SNK returns exactly all the most general sequential nuggets of knowledge in CSD for IM.

4 Related work and discussion

In this paper, we have proposed a definition of sequential association rules and introduced sequential nuggets of knowledge. Those definitions are based on the works presented in [11], but unlike classical sequential pattern mining, our approach focuses on rules with predefined targets as consequents. We have designed SNK, an algorithm based on a pattern-growth strategy (as PrefixSpan [11]) to generate the most general sequential nuggets of knowledge using an interestingness measure that evaluates

the pertinence of a rule. Other efficient works have been proposed for sequential pattern mining. SPADE [15] is as fast as PrefixSpan but uses a bitmap structure which is better adapted to the study of very long sequences but less suitable for short sequences. [9] had proposed a method to generate sequential association rules, but is based on an *a priori*-like strategy with two steps, a candidate test step and a candidate generation step. This approach generates many unnecessary candidates that our pattern-growth approach avoids.
Sequential nuggets of knowledge are defined by a good interestingness measure value. SNK offers the choice between a dozen of interestingness measures. The choice of a suitable measure for a given application domain can be guided by the examination of criteria described in [7] and in [12]. On the other hand, [8] proposes a statistical bootstrap-based method to assess the significance of a measure (thus avoiding false discoveries) that could be used with SNK. A first implementation of SNK is freely available on the web (`http://www.lri.fr/~rance/SNK/`) with some other functionalities.
We envisage to use our algorithm in applications, e.g. on web logs and molecular biology.

5 Acknowledgement

Authors are very grateful to Céline Arnaud for her great help for the implementation of SNK applet. This work was supported in part by the French ACI IMPBio grant RAFALE.

References

[1] Agrawal,R., Imielinski,T., Swami,A.N., (1993) Mining Association Rules between Sets of Items in Large Databases,*Proc. of the 1993 ACM SIGMOD International Conference on Management of Data*, 207–216.

[2] Agrawal,R., Srikant,R., (1995) Mining sequential patterns, *In Proc. Eleventh International Conference on Data Engineering*, 3–14.

[3] Azé,J., Kodratoff,Y., (2002) A study of the Effect of Noisy Data in Rule Extraction Systems, *Proc. of the Sixteenth European Meeting on Cybernetics and Systems Research (EMCSR'02)* (2) 781–786.

[4] Bairoch,A., Apweiler,R., Wu,C.H., Barker,W.C., Boeckmann,.B, Ferro,S., Gasteiger,E., Huang,H., Lopez,R., Magrane,M., Martin,M.J., Natale,D.A., O'Donovan,C., Redaschi,N., Yeh,L.S., (2005) The Universal Protein Resource (UniProt) *Nucleic Acids Res.* 33: D154–159.

[5] Finn,R.D., Mistry,J., Schuster-Backler,B., Griffiths-Jones,S., Hollich,V., Lassmann,T., Moxon,S., Marshall,M., Khanna,A., Durbin,R., Eddy,S.R., Sonnhammer,E.L.L., Bateman,A. (2006) Pfam: clans, web tools and services. *Nucleic Acids Research, Database Issue* 34:D247–D251.

[6] Froidevaux,C.,Lisacek, F.,Rance,B.(2007) Extracting Sequential Nuggets of Knowledge *Proc. of DEXA'07*, 740-750.

[7] Geng, L., Hamilton H.J. (2006) Interestingness Measures for Data Mining: A Survey, *ACM Computing surveys, Vol 38, No3, Article 9*.

[8] Lallich S., Teytaud O. and Prudhomme E. (2006), Association rule interestingness: measure and statistical validation, *in* Quality Measures in data Mining, (Guillet F. and Hamilton H.J. eds.), Springer.

[9] Masseglia,F.,Tanasa,D.,Trousse,B. (2004) Web Usage Mining: Sequential Pattern Extraction with a Very Low Support, *APWeb 2004*, LNCS3007, 513–522.

[10] Nikitin,F., Rance,B., Itoh,M., Kanehisa,M., Lisacek,F. (2004) Using Protein Motif Combinations to Update KEGG Pathway Maps and Orthologue Tables, *Genome Informatics*, 2:266–275.

[11] Pei,J., Han,J., Mortazavi-Asl,B., Wang,J., Pinto,H., Chen,Q., Dayal,U., Hsu,M.-C. (2004) Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach, *IEEE Transactions on Knowledge and Data Engineering*, 16:1424–1440.

[12] Tan,P.N.,Kumar,V., Srivastava,J. (2002) Selecting the Right Interestingness Measure for Association Patterns, *SIGKDD'02*.

[13] Wootton, J.C. and Federhen, S. (1993) Statistics of local complexity in amino acid sequences and sequence databases. *Comput. Chem.* 17:149–163.

[14] Yun H., Ha D., Hwang B. and Ryu K.H. (2003), Mining association rules on significant rare data using relative support, *The Journal of Systems and Software* 67 (2003), 181-191.

[15] Zaki,M.J. (2001) SPADE: An Efficient Algorithm for Mining Frequent Sequences, *Machine Learning Journal, special issue on Unsupervised Learning (Doug Fisher, ed.)*, 42:31–60.

[16] Zhang, T. (2000) Association Rules. In T. Terano, H. Liu, A.L.P. Chen (Eds), *Proceding of PAKDD 2000, LNAI 1805*, 245–256, Springer-Verlag, 2000.