Extracting Sequential Nuggets of Knowledge

Froidevaux Christine[1]**, Lisacek Frédérique[2], and Rance Bastien[1]

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

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 and apply it to an example of molecular biology. Unexpected nuggets that are produced may help scientists refine a rough preliminary classification. A first implementation in Java is freely available on the web[3].

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 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 paper, 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

** to whom correspondence should be addressed
[3] http://www.lri.fr/∼rance/SNK/

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 paper, we present the algorithm SNK which calculates the most general Sequential Nuggets of Knowledge and illustrate its use in the domain of molecular biology, more specifically, in the perspective of protein functional classification. 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. This approach is particularly interesting in biology.

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 show in (section 4) how this algorithm is useful in an example of SNK application in the domain of molecular biology. We report related work and conclude by discussing our results and giving some perspectives (section 5).

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 interestingness measure value (cf. section 2.2) 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 be considered as important, even if $CONS$ is often found without ANT. 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: $s-\text{postfix}(CSD) = \{(\alpha/s),\ \alpha \in CSD\}$
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) $\text{support}_{s_1.s_2-\text{postfix}(CSD)}(r) = \text{support}_{CSD}((s_1 \cdot s_2 \cdot ant(r)) \rightarrow cons(r))$
(iii) $\text{support}_{CSD}(r) \geq \text{support}_{s_1-\text{postfix}(CSD)}(r)$.

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. $\not\exists r_1, r_2 \in RESULTS$ 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.
measure; // measure$_{IM,CSD}$(r) evaluates the value of r for IM in CSD
support: // support$_S$(r) evaluates the support of r in S
Begin SI = the set of all items of I occurring in elements of S;
Foreach x in SI **do**
if support$_S(x \rightarrow y) \geq min_supp$ **then**
if 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,$p \cdot x$,$RESULTS$)
endifendifendifendfor end_SNKrec;

Running example:
Let $min\_supp = 2$, IM = confidence, $min\_meas = 1$. SNK yields the set of all the maximal sequential nuggets of knowledge:
$RESULTS = \{\langle e, e\rangle \rightarrow tg_2, \langle e, a\rangle \rightarrow tg_2, \langle c, b\rangle \rightarrow tg_2, \langle c, a\rangle \rightarrow tg_2, \langle g, f\rangle \rightarrow tg_2, \langle b, c\rangle \rightarrow tg_1, \langle f, g\rangle \rightarrow tg_1, \langle a, c\rangle \rightarrow tg_1\}$.

3.2 Properties of SNK

First the algorithm is sound and complete w.r.t its specification [6]. Formally:

Theorem 2 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. Then:

SNK returns exactly all the most general sequential association rules r on CSD that satisfy $supp_{CSD}(r) \geq min\_supp$ and $meas_{IM,CSD}(r) \geq min\_meas$.

The time complexity of SNK is related to the number of target items, and for each target item, to the number of recursive calls of SNKrec. 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. Each call requires a calculation of support and of IM measure, and involves either the cost of a postfix-projection or that of the add_rule method. With our depth-first search approach all the projected databases need not be stored in memory and they can be built independently. The analysis shows (see [6] for details) that the theoretical time complexity is 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.

SNK allows to discover rules describing regularities in a sequential data set. Moreover, SNK provides the user with a parameterisation process for adapting the tool to specific needs. The user can select among a dozen measures the measure that best fits his application field (by default confidence is selected) [7]. A bootstrap mode is also available, where SNK is run on a categorised sequential database resampled from the original database as an input in order to check the consistency of the generated rules. In the data mining mode, SNK runs in about 3 seconds for mining sequential nuggets of knowledge for 760 tuples (described by sequences of size less than 17 where the set of items has about 35 distinct elements), 6 seconds for 1200 tuples. SNK is fully implemented in Java and the web Applet is freely available on SNK website (http://www.lri.fr/∼rance/SNK/).

4 Example

We show how SNK can be useful through the study of a family of bacterial proteins. Each protein is described by its sequence of motifs (we call "motif" a functional or well conserved part of the amino acid sequence). We consider the Phospholipase D (PLD) family of proteins which are present in all species from virus to eukaryote, and involved in many cell processes. These proteins are grouped together simply because they carry the PLDc motif repeated once. They also contain a wide range of other motifs. In [10], a surprising regularity concerning the C-terminal part of proteins was reported. More precisely, the distance between the end of the second PLDc motif and the C-terminal end of the protein (rightmost) was shown to correlate with the known functions of the proteins. Consequently, proteins could be grouped into classes using this distance as a classification criterion. In the remainder of this section we will refer to the length of this region as the *C-terminal length* (this length is either: 40, 60, 72, 82, 100). Each class is then functionally consistent. Using SNK we have investigated a possible relationship between module architecture, C-terminal length and function. We have considered all bacterial proteins of the UniProtKB database [4] which contain two PLDc motifs. The corresponding set of proteins showed a variety of motif combinations involving other protein family signatures as well as

so-called "low complexity regions" (poorly informative sequences [13]). The total number of proteins is 676. We first considered the possible existence of a link between low-complexity regions and C-terminal length. In this first test, proteins were described as successions of PLDc motifs and low complexity regions. We studied a set of proteins containing all the PLD proteins with C-terminal length from classes "72" and "82" using Zhang's interestingness measure. SNK was performed with a very low support threshold (min_supp=15) and with a good measure threshold (min_meas=0.8). Among the 20 most general sequential nuggets of knowledge obtained, 3 rules were especially interesting. In the rules presented below, *lc* denotes low-complexity region and the values between brackets are respectively support and Zhang's measure values.

`(1a) lc,PLDc,PLDc -> 82, (273,0.80)`,
`(1b) lc,lc,PLDc -> 82, (174,0.68)`,
`(1c) PLDc,PLDc,lc -> 72, (136,0.90)`

The sequential association rules returned by SNK are high quality rules. Rule (1a) and (1c) highlight the importance of the order between modules in the assignment to a class. The location of low complexity regions is closely linked to the C-terminal length. Depending on whether *lc* is in front of or behind the double PLDc motif, the conclusion of the rule is one or the other class. Simple association rules could not have expressed such a clear distinction.

In a second test information about protein family signatures as compiled in both Pfam-A [5] and Pfam-B databases was added. Amongst the rules generated with min_supp=7 and min_meas=0.9,

`(2a) PLDc,Pfam-B_115,Pfam-B_2786 -> 40, (7,0.90)` and
`(2b) PLDc,Pfam-B_115,Pfam-B_6054 -> 40, (7,0.93)`

are the only rules where PLDc precedes Pfam-B_115 and therefore appear to characterise class 40. In all other rules where the two entities occur Pfam-B_115 precedes PLDc. A complementary test was performed with the same initial data set but taking the protein function as a target for SNK (either diacyltransferase, cardiosyntase, transphosphatidylase or unspecialised phospholipase D). Amongst the 9 rules (min_supp=7, min_meas=0.9), one strongly corresponds to the cardiosyntase function:

`(3a) Pfam-B_1038,lc,Pfam-B_115,Pfam-B_2786 -> cardio, (7,1.00)`

This rule appears quite similar to one of the rules generated (same thresholds) for the length criterion

`(2c) lc,Pfam-B_115,lc,Pfam-B_2786 -> 60, (15,0.94)`

Likewise, `(3b) lc,Pfam-B_5151 -> diacyltransferase, (7,0.91)` strongly corresponds to the diacyltransferase function while `(3c) Pfam-B_5151 -> 72, (53,1.00)` was previously generated for the length criterion.

This generalises the correlation suggested in [10] between length 60 and 72 respectively and the cardiosyntase and diacyltransferase functions. Other rules generated with the protein function as a target are potentially misleading due to inconsistencies of the automated assignment of function in these proteins. We are currently testing the possibility of correcting mistakes using rules generated with the length criterion.

5 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 with some other functionalities.
Finally we have presented an example in biology involving the PLDc family of proteins. The link between C-terminal length of a PLDc protein and its function was investigated. Let us recall that a protein function usually corresponds to a specific sequence of structural units. Most studies take into account the combinatorial aspect of the structural composition of proteins. We showed that the identification of sequential constraints could lead to a refinement of the functional classification of proteins. As a result, a large class grouped upon one rough criterion can be subdivided into sub-classes upon explicit and informative distinctive traits. We are currently testing the possibility of using the rules discovered as a way of automatically correcting mistakes.
We also envisage to use our algorithm in other applications, e.g. on web logs, and to extend it by adding non-sequential items in the antecedent of a rule. In that way, it could take into account more expressive descriptions of objects. Since the projected databases can be considered independently, we also plan to develop a distributed version for a cluster of PC thereby drastically speeding up SNK.

6 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) Mining sequential nuggets of knowledge *UPS-LRI, Technical report*, to appear.

[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, to appear *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.