Information Systems 34 (2009) 438–453

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

Non-redundant sequential rules—Theory and algorithm

David Lo [a,*], Siau-Cheng Khoo [b], Limsoon Wong [b]

[a] School of Information Systems, Singapore Management University, Singapore
[b] Department of Computer Science, National University of Singapore, Singapore

ARTICLE INFO

Article history:
Received 8 January 2009
Accepted 21 January 2009
Recommended by: D. Shasha

Keywords:
Theoretical data mining
Frequent pattern mining
Sequential pattern mining
Sequential rules
Non-redundant rules

ABSTRACT

A sequential rule expresses a relationship between two series of events happening one after another. Sequential rules are potentially useful for analyzing data in sequential format, ranging from purchase histories, network logs and program execution traces.

In this work, we investigate and propose a syntactic characterization of a non-redundant set of sequential rules built upon past work on compact set of representative patterns. A rule is redundant if it can be inferred from another rule having the same support and confidence. When using the set of mined rules as a composite filter, replacing a full set of rules with a non-redundant subset of the rules does not impact the accuracy of the filter.

We consider several rule sets based on composition of various types of pattern sets—generators, projected-database generators, closed patterns and projected-database closed patterns. We investigate the completeness and tightness of these rule sets. We characterize a tight and complete set of non-redundant rules by defining it based on the composition of two pattern sets. Furthermore, we propose a compressed set of non-redundant rules in a spirit similar to how closed patterns serve as a compressed representation of a full set of patterns. Lastly, we propose an algorithm to mine this compressed set of non-redundant rules. A performance study shows that the proposed algorithm significantly improves both the runtime and compactness of mined rules over mining a full set of sequential rules.

1. Introduction

Sequential pattern mining first proposed by Agrawal and Srikant [1] has been the subject of active research [2–7]. Given a database containing sequences, sequential pattern mining identifies sequential patterns appearing with enough support. It has potential application in many areas such as analysis of market data, purchase histories, web logs, etc.

Sequential rules express temporal relationships among patterns [8]. It can be considered as a natural extension to sequential patterns, as association rules are to frequent itemsets [9]. A sequential rule expressed as $pre \rightarrow post$, specifies that there is sufficiently high confidence that the pattern *post* will occur in sequences following an occurrence of *pre*. Compared to sequential patterns, rules allow better understanding of temporal behaviors exhibited in a sequence database. Consider a classic example of purchasing behavior in a video shop [1]: a customer who buys Star Wars episode IV will likely buy episodes V and VI in the future. The purchase pattern $\langle IV, V, VI\rangle$ is the pattern showing the purchase behavior. However, imagine a standard video shop with hundreds of buyers with various preferences. The pattern $\langle IV, V, VI\rangle$ will tend to occur with a low support. Mining with low support will return the pattern, however, typically along with many irrelevant or spurious patterns. Rules can throw away

* Corresponding author. Tel.: +65 98421014.
E-mail addresses: davidlo@smu.edu.sg (D. Lo), khoosc@comp.nus.edu.sg (S.-C. Khoo), wongls@comp.nus.edu.sg (L. Wong).

0306-4379/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.is.2009.01.002

many spurious patterns by introducing the notion of confidence to the set of patterns. Only rules satisfying both support and confidence thresholds are mined.

Sequential rules extend the usability of patterns beyond the understanding of sequential data. A mined rule represents the *constraint* that its premise is followed by its consequent in sequences. Hence, rules are potentially useful for detecting and filtering anomalies which violate the corresponding constraints. They have applications in detecting errors, intrusions, bugs, etc. Mining rule-like sequencing constraints from sequential data has been shown useful in medicine (e.g., [10]) and software engineering (e.g., [11–13]) domains. Some examples of useful rules include:

1. (Market data) If a customer buys a car, he/she will eventually buy car insurance. This is potentially useful in designing personalized marketing strategy.
2. (Medical data) If a patient has a fever, which is followed by a drop in thrombosite level and followed by appearance of red spots in the skin, then it is likely that the patient will need a treatment for dengue fever. This is potentially useful in predicting a suitable type of treatment needed for a patient.
3. (Software data) If a Windows device driver calls *KeAcquireSpinLock*, then it eventually needs to call *KeReleaseSpinLock* [14].

Spiliopoulou [8] proposes generating a *full set* of sequential rules (i.e., all frequent and confident rules) from a *full set* of sequential patterns (i.e., all frequent patterns). Generating a full set of sequential rules can be very expensive. The number of frequent patterns is exponential to the maximum pattern length: if a sequential pattern of length l is frequent, all its $\mathcal{O}(2^l)$ subsequences are frequent as well. Each frequent pattern of length l can possibly generate $l-1$ rules (depending on the minimum confidence threshold). Hence, there is an exponential growth in the number of rules with respect to the maximum pattern length.

To tame the explosive growth of rules, we propose mining a *non-redundant* set of sequential rules. Central to our method is the notion of *rule inference*. This notion is used to define and remove redundancy among rules. When using the set of mined rules as a composite filter, replacing a full set of rules with the non-redundant subset of rules does not impact the accuracy of the filter.

There have been many studies on mining frequent sequential patterns [1–5,15–17]. These studies include those mining a compact representation of patterns, referred to as closed patterns [6,7] and generators [18,19]. These compact representative patterns can be mined with much more efficiency than the full set of frequent patterns. However, there *has not been* any study relating these compact representative patterns with a non-redundant set of sequential rules. In particular the following questions need to be addressed: Can a non-redundant set of rules be obtained from compact representative patterns? What types of compact representative patterns need to be mined to form non-redundant rules? What do we mean by a non-redundant set of rules? Can we characterize the non-redundant set of rules? How to use representative patterns to form non-redundant rules? How much effort is needed to obtain a non-redundant set of rules from compact representative patterns? Can we design an efficient algorithm to obtain a non-redundant set of rules from patterns?

In this paper, we address the above research questions. We focus on performing an investigation and a characterization of a set of non-redundant sequential rules built upon existing studies on compact sets of representative sequential patterns. In addition, we propose an algorithm, develop a tool, and perform a performance study on mining a non-redundant set of sequential rules.

We investigate four different sets of patterns, namely generators, projected-database generators, closed patterns and projected-database closed patterns. For the projected-database generators and closed patterns, aside from the format and support values of patterns, we also consider their projected database (cf., [3,6]).

A rule set can be formed by composing patterns. We investigate various configurations of compositions of the above four sets of patterns. These sets are then evaluated based on the two criteria of completeness and tightness. A rule set is complete, if each frequent and confident rule can be inferred by one of the rules in the rule set. A rule set is tight, if the set contains no redundant rules. We characterize a tight and complete set of non-redundant rules based on these configurations.

Additionally, to further reduce the number of mined rules, we propose a rule compression strategy to compress the set of non-redundant rules. This strategy is in the same spirit as how closed patterns are used as a compressed representation of a full set of frequent patterns.

We propose an algorithm to mine this compressed set of non-redundant rules. Our performance study shows much benefit in mining non-redundant rules over a full set of rules. The study shows that the runtime and number of rules mined can be reduced by up to 5598 *times* and 8583 *times*, respectively!

The contributions of our work are as follows:

1. We propose a concept of non-redundant rules based on logical inference.
2. We investigate different sets of patterns and their various compositions to form different sets of rules. We study the quality of these rule sets with respect to completeness and tightness.
3. We characterize a tight and complete set of non-redundant rules based on compositions of patterns.
4. We propose and characterize compression of the non-redundant set of rules.
5. We develop an algorithm to mine the compressed set of non-redundant rules and show that it performs much faster than mining a full set of sequential rules.

The outline of this paper is as follows. Section 2 presents terminologies and definitions used. Among other things