Vandana Publications
IJEMR

Volume-5, Issue-3, June-2015
International Journal of Engineering and Management Research
Page Number: 505-508

A Newfolder for Signature based Indexing Method

Subba Lakshmi S[1], Vinay Kumar S[2], Asha Rani K[3]
[1,2,3]Assistant Professor, Department of CSE, GPREC, Nandyal, INDIA

ABSTRACT

A number of algorithms have been proposed for the discovery of data's from the large database. However, since the number of generated patterns can be large, selecting which patterns to analyze can be nontrivial. There is thus a need for algorithms and tools that can assist in the selection of discovered patterns so that subsequent analysis can be performed in an efficient and, ideally, interactive manner. In this project, we propose a signature-based indexing method to optimize the storage and retrieval of a relative data's from the large database.

***Keywords*----** Data Mining, OODB, Index, Retrieval

I. INTRODUCTION

Many rule discovery algorithms in data mining generate a large number of patterns/rules, sometimes even exceeding the size of the underlying database, with only a small fraction being of interest to the user. It is generally understood that interpreting the discovered patterns/rules to gain insight into the domain is an important phase in the knowledge discovery process. However, when there are a large number of generated rules, identifying and analyzing those that are interesting becomes difficult. For example, providing the user with a list of association rules ranked by their confidence and support might not be a good way of organizing the set of rules as this method would overwhelm the user and not all rules with high confidence and support are necessarily interesting for a variety of reasons.

Therefore, to be useful, a data mining system must manage the generated rules by offering flexible tools for rule selection. In the case of association rule mining, several approaches for the post processing of discovered association rules have been discussed. One approach is to group "similar" rules which work well for a moderate number of rules. However, for a larger number of rules it produces too many clusters. A more flexible approach is to allow the identification of rules that are of special importance to the user through templates or data mining queries. This approach can complement the rule grouping approach and has been used to specify interesting and uninteresting classes of rules (for both association and episodic rules).

The importance of data mining queries has been highlighted by the introduction of the inductive database concept, which allows the user to both queries the data and query patterns, rules, and models extracted from these data.

II. RELATED WORKS

The temporal patterns described in this paper consist of two components: a set of states and a set of relationships between those states that represent the order of states within the pattern. In order to retrieve such patterns efficiently, any indexing method should deal with both temporal concepts states and state relationships.

The problem of indexing has been studied in depth in the database literature, (for example, B+ trees, R trees etc. However, studies on set-based indexes that support queries on set-valued attributes (i.e., attributes that are sets of items) are limited. Apply the signature file technique to support the processing of queries involving set-valued attributes in OODBs. Two signature file organizations are considered: the sequential signature file and the bit-slice file. The bit-slice approach still needs to examine every signature in the file but only a part of it. In order to avoid reading every signature in the signature file, the hierarchical file organization uses several levels of signatures. The higher levels perform coarse filtering before the signatures on the lower levels are consulted. Examples of the hierarchical file organization include the S-tree and the SG-tree. The partitioned file organization approach avoids reading every signature by grouping the signatures into several partitions such that all signatures in a given partition possess the same component part, called the signature key. The signature key used is usually a substring of the signature. By partitioning the signatures, some of the partitions need not to be searched during the execution of a query so that the number of accesses can be reduced.

III. PRELIMINARIES

3.1 Problem Description

Definition 1 (state sequence). Let S denote the set of all possible states. A state s 2 S that holds during a period of time [b, f) is denoted as (b, s, f), where b is the start time, and f is the end time. The (b, s, f) triple is termed a state interval. A state sequence on S is a series of triples defining state intervals (b1, s1, f1), h (b2, s2, f2). . . (bn, sn, fn)i, where bi _ bi)1, and bi < fi.

Definition 2 (temporal pattern). Given n state intervals (bi, si, fi), 1 _ i _ n, a temporal pattern of size n is defined by a pair (s,M), where s : f1, . . . , ng ! S maps index i to the corresponding state, andM is an n _ n matrix whose elements M½i, j_ denote the relationship between intervals ½bi, fi) and ½bj, fj). The size of a temporal pattern _ is the number of intervals in the pattern, denoted as dim (_). If the size of _ is n, then _ is called an n-pattern.

If the state intervals within the state sequences have been ordered in increasing index according to their start times, end times, and states, the resulting temporal patterns are considered normalized, and only seven out of 13 Allen relationships [32] are required, namely, before (b), meets (m), overlaps (o), is-finished-by (fi), contains (c), equals (¼), and starts (s).1 For the rest of this paper, let Rel ¼ f¼, c, fi, s, o,m, bg be the set of these seven relationships. A normalized temporal pattern does not contain temporal extensions because it has been abstracted from the state intervals in a specific state sequence. Fig. 1 shows four normalized temporal patterns defined over a set of states S ¼ fA, B,C,Dg and a set of interval relations Rel.

Definition 3 (subpattern). A temporal pattern _ ¼ (s_,M_) is a subpattern of _ ¼ (s_,M_), denoted _ v _, if dim(s_,M_) _ dim(s_,M_) and there is an injective mapping _ : f1, . . . , dim(s_,M_)g ! f1, . . . , dim(s_,M_)g such that 8i, j 2 f1, . . . , dim(s_,M_)g : s_(i) ¼ s_(_(i)) ^M_½i, j_ ¼ M_½_(i), _(j)_: Informally, it can be stated that a pattern _ is a subpattern of _ if _ can be obtained from _ by removing intervals. As an example, consider the patterns in Fig. 1, p1 is a subpattern of p3, but it is not a subpattern of p4. We can obtain p1 from p3 by removing interval state D, on the other hand, removing interval states C and D from p4 would not result in p1.

Definition 4 (content-based queries). Let D be a temporal pattern database and q be a query pattern. The four forms of content-based queries that this research supports include the following:

1. Sub pattern queries. Find those patterns in D that contain q.
2. Super pattern queries. Find those patterns in D that are a sub pattern of q.
3. Equality queries. Find those patterns in D equal to q.
4. K-nearest sub pattern queries. Find the k most similar patterns in D to q.

Superpattern queries are useful when searching for the characteristic parts of a large pattern, while k-nearest subpattern queries limit the number of patterns generated by subpattern or superpattern queries. The problem of content-based retrieval of temporal patterns can be formally defined as follows:

Given a database D of discovered temporal patterns, describe a processing technique that allows the user to find efficiently temporal patterns in D that satisfy the content-based queries above.

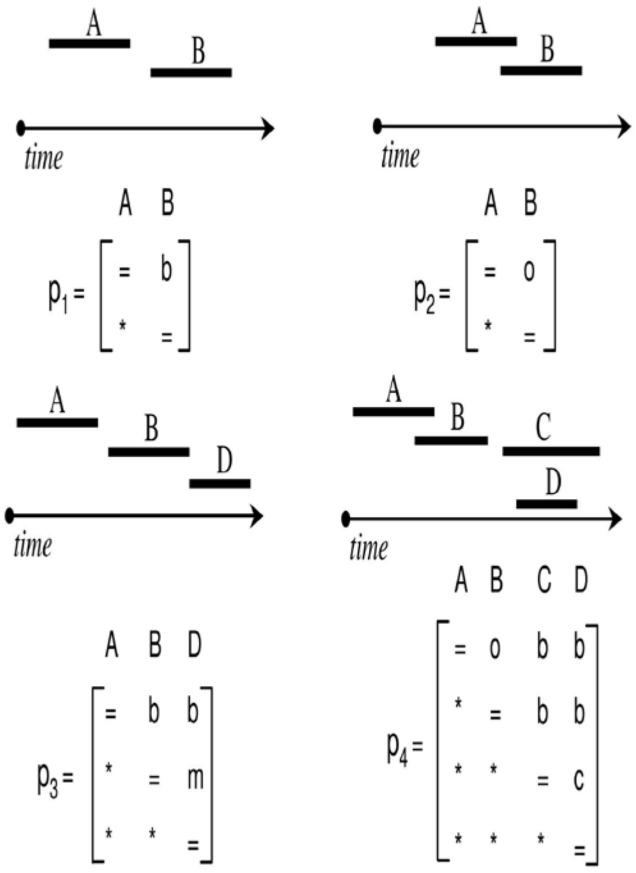


Fig. 1. Example of temporal patterns

3.2 Temporal Pattern Similarity

To answer k-nearest sub pattern queries, a suitable measure of similarity among temporal patterns needs to be defined. To do so, three properties must be considered:

1. Temporal patterns are variable-length objects that cannot be represented in a k-dimensional metric space.
2. Each pattern contains a list of states.
3. Each pattern contains a set of state relationships.

IV. SIGNATURE-BASED INDEX FOR RETRIEVAL OF TEMPORAL PATTERNS

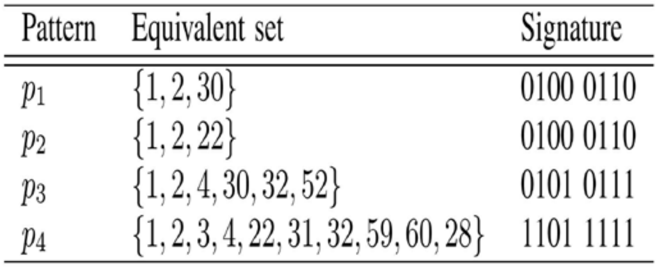
This section first describes the method used to construct the signature-based index from a collection of temporal patterns and to answer a query using the index. In order to facilitate this discussion, a brief overview of the use of signature files in set retrieval is provided, and the new Terminologies used are introduced.

4.1 Signature Files

A signature is a bit pattern formed for a data object, which is then stored in the signature file. Signature files were originally proposed in the context of word retrieval in text databases. Recently, they have been used in a wider set of applications, including office automation, hypertext systems, relational and object-oriented databases, and data mining. This brief review of signature files focuses on their use in facilitating the retrieval on set-valued attributes. Given that T and Q denote the target set and query set, respectively, three commonly used set-valued queries are

1. Subset query (T _ Q). The target set is a superset of the query set.
2. Superset query (T Q). The target set is a subset of the query set.
3. Equality query (T = Q). The target set is equal to the query set.

An initial target signature is generated for each target set as follows: Each element in a target set is hashed to a bit pattern termed an element signature. All element signatures are of bit length F, and exactly b bits are set to "1," where b < F. F is termed the length of a signature, while b is termed the weight of an element's signature. A target signature is obtained by the bitwise union of all element signatures. The pairing of a target signature and an identifier of the object containing that target set is stored in the signature file.



Pattern	Equivalent set	Signature
p_1	$\{1,2,30\}$	0100 0110
p_2	$\{1,2,22\}$	0100 0110
p_3	$\{1,2,4,30,32,52\}$	0101 0111
p_4	$\{1,2,3,4,22,31,32,59,60,28\}$	1101 1111

TABLE 1

Equivalent Sets and Signatures of Temporal Patterns.

In set retrieval using signature files, set-valued queries are processed in the following way. A query signature is generated from the query set Q (in the same way as the target signature). During the next step, the filtering step, each signature in the signature file is examined with the query signature and becomes a drop (a candidate that may satisfy the query) if it satisfies a predefined condition as follows:

1. T superset of Q: target signature ^ query signature ¼ query signature.
2. T subset of Q: target signature ^ query signature ¼ target signature.
3. T = Q: query signature = target signature.

In the next step, false-drop resolution, each candidate is retrieved and examined to see if it actually satisfies the query condition. Candidates that fail the test are termed false drops, while successful candidates are termed actual drops. False drops occur due to the collision of element signatures and the superimposed coding method. False drops affect the number of block accesses (I/O time), as well as the processing time used to decide whether a target set should be returned to the user. The main signature file issue is therefore the proper control of false drops.

4.2 Constructing Signature Files for Temporal Patterns

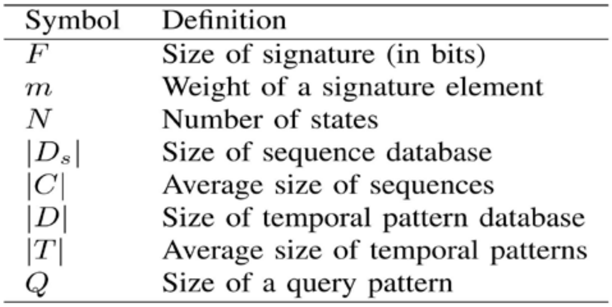
The signature of a temporal pattern is created by converting the temporal pattern into an equivalent set from which the signature is generated.

V. EXPERIMENTS

To assess the performance of the proposed methods, the evaluate Sub Pattern, evaluate Super Pattern, and evaluate Equality were implemented on SSF and BSSF signature files. In addition, sequential versions of the methods (SEQ) were also implemented as baseline methods, which process queries by sequentially retrieving the target pattern (without using an index) from the database and comparing it against the query pattern. All programs are written in Java Language. The experiments were conducted on synthetic data sets on a 1.3-GHz Intel Celeron PC with 384 Mbytes of RAM running Windows XP Professional.

The following sections show the performance of evaluates SubPattern and evaluate SuperPattern for processing subpattern and superpattern queries, respectively. Update cost is not considered as it is assumed that the index is only created once after frequent patterns have been generated by a data mining process. Experimental parameters are listed in Table 2.

First, the size of temporal pattern t was determined randomly from a Poisson distribution with a mean equal to jTj. Then, a temporal pattern of size t is randomly picked from the set of frequent temporal patterns and added to the database D.



Symbol	Definition
F	Size of signature (in bits)
m	Weight of a signature element
N	Number of states
$\|D_s\|$	Size of sequence database
$\|C\|$	Average size of sequences
$\|D\|$	Size of temporal pattern database
$\|T\|$	Average size of temporal patterns
Q	Size of a query pattern

TABLE 2 Parameters

VI. CONCLUSIONS AND FUTURE WORK

The use of a signature-based index for content-based retrieval of temporal patterns has been presented. The signatures of temporal patterns are created by first converting temporal patterns into equivalent sets and then generating the signatures from the equivalent sets. The study focused on the sequential and BSSF organizations and a series of experiments compared the performance of both signature files in processing subpattern and superpattern queries.

In conclusion, the use of signature files improves the performance of temporal pattern retrieval. The bit-slice signature file performs better than the SSF and is a good choice for content-based retrieval of temporal patterns. This retrieval system is currently being combined with visualization techniques for monitoring the behavior of a single pattern or a group of patterns over time.

REFERENCES

[1] T. Imielinski and A. Virmani, “Association Rules . . . and what’s Next? Towards Second Generation Data Mining Systems,” Proc. Second East European Symp. Advances in Databases and Information Systems (ADBIS ’98), pp. 6-25, 1998.

[2] L. Geng and H.J. Hamilton, “Interestingness Measures for Data Mining: A Survey,” ACM Computing Surveys, vol. 38, no. 3, 2006.

[3] H. Toivonen, M. Klemettinen, P. Ronkainen, K. Hatonen, and H. Mannila, “Pruning and Grouping of Discovered Association Rules,” Proc. ECML Workshop Statistics, Machine Learning, and Knowledge Discovery in Databases, pp. 47-52, 1995.