Korean Journal of Remote Sensing, Vol.22, No.5, 2006, pp.457~462

# Implementation of Subsequence Mapping Method for Sequential Pattern Mining

**Nguyen Thu Trang*, Bum Ju Lee*, Heon Gyu Lee*, Jeong-Seok Park**, and Keun Ho Ryu*†**

Database and Bioinformatics Laboratory, Chungbuk National University, South Korea*
College of Electrical, Electronic & Information Engineering, ChungJu National University, South Korea**

**Abstract :** Sequential Pattern Mining is the mining approach which addresses the problem of discovering the existent maximal frequent sequences in a given databases. In the daily and scientific life, sequential data are available and used everywhere based on their representative forms as text, weather data, satellite data streams, business transactions, telecommunications records, experimental runs, DNA sequences, histories of medical records, etc. Discovering sequential patterns can assist user or scientist on predicting coming activities, interpreting recurring phenomena or extracting similarities. For the sake of that purpose, the core of sequential pattern mining is finding the frequent sequence which is contained frequently in all data sequences. Beside the discovery of frequent itemsets, sequential pattern mining requires the arrangement of those itemsets in sequences and the discovery of which of those are frequent. So before mining sequences, the main task is checking if one sequence is a subsequence of another sequence in the database. In this paper, we implement the subsequence matching method as the preprocessing step for sequential pattern mining. Matched sequences in our implementation are the normalized sequences as the form of number chain. The result which is given by this method is the review of matching information between input mapped sequences.

Key Words : Motif, Protein sequence, Subsequence, Sequential Pattern Mining.

## 1. Introduction

Data mining is a technology that blends traditional data analysis methods with sophisticated algorithms for processing large volumes of data. It has also opened up exciting opportunities for exploring and analyzing new types of data and for analyzing old types of data in new ways. All data mining algorithms attempt to fit a model to the data. They examine the data and determining a model that is closest to the characteristics of the data being examined. Nowadays, the rapid growth of the amount of stored digital data and the recent development data mining techniques lead to increase in methods for the exploration of data, creating new data mining problems and solution.

Sequential Pattern Mining is the mining approach which approach which addresses the problem of

Received 15 September 2006; Accepted 20 Octorber 2006.

s† Corresponding Author: K. H. Ryu (khryu@dblab.chungbuk.ac.kr) discovering the existent maximal frequent sequences in a given databases. This technique is applied to broad applications included the analysis of customer purchase patterns or Web access pattern, analysis of DNA and protein sequences··· Algorithms for this problem are relevant when the data to be mined has some sequence nature such as events in the case of temporal information or amino-acid sequences in bioinformatics.

The sequential pattern mining problem was first produced by [R. Agrawal and R. Srikant, 1995]. The input data is a sequence data set. Each sequence in data set is a list of transactions, where each transaction is a set of items. The problem is to find all sequential patterns (or frequent pattern) with a user-specified minimum support, where the support of a *sequential pattern is the percentage of sequences in* data set that contain the pattern. The substance of counting the support of one sequence is to check if this sequence is contained by another sequence. If the result is true, this sequence is called subsequence and its support count is increased 1. The checking process is simple if the sequences have one transaction with individual items. But in fact, the sequences in data set can have many transactions and each transaction also has many items. In this case, to check subsequences consists of checking the order of items in each transaction of that sequence. So sequential pattern discovery is a computationally challenging task because there are exponentially many sequences contained in a given data sequence.

The core of sequential pattern mining is finding the *frequent sequence which is contained frequently in all* data sequences. Beside the discovery of frequent itemsets, sequential pattern mining requires the arrangement of those itemsets in sequences and the discovery of which of those are frequent. So before mining sequences, the main task is checking if one sequence is a subsequence of another sequence in the database. In this paper, we implement the subsequence matching method as the preprocessing step for sequential pattern mining with protein sequences in bioinformatics. Matched sequences in our implementation are the normalized sequences as the form of number chain.

The remainder of the paper in organized as follows: Section 2, the sequential pattern mining problem is defined and the common sequential pattern mining algorithms are illustrated. In section 3, our approach is described with protein sequence databases and the implementation of this system with its user-interface figs. will be presented in the fourth section. Finally, in section 5, some conclusions and the outline directions for future work will be presented.

## 2. Related Work

In this section, the problem of sequential pattern mining is defined and the common algorithms which are introduced for solving it will be summarized.

### 1) Problem definition

Let I = $\{i_1, i_2 \cdots i_k\}$ be a set of all items. An itemset is a non-empty subset of items. A sequence which is an ordered list of transactions (elements) can be denoted as $s = (s_1\ s_2 \cdots s_n)$ where each element sj is a collection of one or more event as $s_j = (i_1, i_2 \cdots i_k)$. The number of elements in a sequence s is called the length of the sequence and a sequence with length k is usually called *k-sequence*. A sequence is maximal if it is not contained in any other sequence. A sequence $t$ is a subsequence of another sequence $s$ if each ordered element in t is a subset of an ordered element in s. Formally, the sequence $t = (t_1 t_2 \cdots t_m)$ is a subsequence of $s = (s_1 s_2 \cdots s_n)$ if there exists integers $1 \leq j_1 < j_2 < \cdots < j_m \leq$ n such that $t_1 \subseteq s_{j1}$, $t_2 \subseteq s_{j2}, \cdots$, $t \subseteq s_{jm}$. If $t$ is a subsequence of $s$, then we say $t$ is contained in $s$. Table 1 gives some examples

Table 1. Example of subsequence.

| Sequence s | Sequence t | Is t contained in s? |
|---|---|---|
| <{2,4} {3,5,6} {8}> | <{2} {3,6}> | YES |
| <{1,2} {3,4}> | <{1} {2}> | NO |

illustrating the idea of subsequences for various sequences. As in the example table, the sequence <{2} {3,6}> is the subsequence of <{2,4} {3,5,6} {8}> since $\{2\}\subseteq\{2,4\}$; $\{3,6\}\subseteq\{3,5,6\}$.

A *sequence database* S is a set of tuples <*sid, s*> where *sid* is a sequence_id and *s* is a sequence. A tuple <*sid, s*> is said to contain a sequence $\alpha$ if $\alpha$ is a subsequence of *s*. The *support* of a sequence $\alpha$ is the number of tuples in the database containing$\alpha$. The sequential pattern mining problem can be defined as: *Given the sequence database S and a positive integer min-support as the support threshold, sequential pattern mining is to find all frequent subsequence ? whose occurrence frequency in S is no less than min-support that means* $support_S(\alpha) \geq$ *min-support.*

### 2) Sequential pattern mining algorithms

A number of algorithms and techniques have been proposed to deal with the problem of sequential pattern mining. They are divided into two common approaches such as *apriori-based* and *pattern-growth* algorithms which are being used as the basis for other structures pattern mining algorithms.

The Apriori-based method is the generation - and - test approach based on priori principle: *If k-sequence is frequent, then all of its subsequence (k-1)-sequence must also be frequent.* The first algorithm as following Apriori-based approach was AprioriAll [R. Agrawal and R. Srikant, 1995]. It is a three-phase algorithm: first finding all itemsets with min-support (frequent itemsets), after that transforming the database so that each transaction is replaced by the set of all frequent itemsets contained in the transaction, and final finding sequential pattern. The GSP (Generalized Sequential Pattern) algorithm [Srikant, R. and Agrawal, R, 1996] is an evolution of AprioriAll, allowing for the incorporation of gap constraints. The key difference between these two algorithms resides on the candidate generation produce: GSP creates a new candidate whenever the prefix of a sequential is equal to suffix of another one; while the candidate generation of AprioriAll works by joining two frequent (k-1)-sequences when their maximal prefixes are equal. To reduce the database scanning times, SPADE (Sequential PAttern Discovery using Equivalence classes) algorithm was introduced in [Zaki M., 1998]. It successfully finds all frequent sequences in only three database scans. The pattern-growth method which is the partition-based, divide and conquers approach is a more recent approach to deal with sequential pattern mining problems. The key idea is to avoid the candidate generation step altogether and to focus the search on a restricted portion of the large database. Instead of repeatedly scanning the entire database, it recursively project a sequence database into a set of patterns mined and after that mines locally frequent patterns in each project database. Based on this philosophy, the first introduced pattern-growth algorithms is FreeSpan [J. Han, 2000], and then the improved method is PrefixSpan [J. Han, 2001]. Both methods generate projected databases, but they differ at the criteria of database projection: FreeSpan creates projected databases based on the current set of frequent patterns without a particular ordering, whereas PrefixSpan projects database by growing frequent prefixes.

## 3. Materials and Method

### 1) Materials

In area like bioinformatics, which exhibit limited

Table 2. Example of CATH database.

| Protein ID (PDB CATH code) | Protein sequence (Homology H-level) | CATH code |
|---|---|---|
| 1fupA02 | SSNDVFPTAMHVAAL··· | 1.20.200.10 |
| 1go3E01 | YELIEGEVVDVVEFGS··· | 2.40.50.140 |
| 1ubaA00 | QEKEAIERLKALGFPE··· | 1.10.8.10 |

Table 3. Example of Motifs in Protein.

| Location in seq. | Motif | | | | | | | Protein |
|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
| 14 | G | V | S | A | S | A | V | Ka RbtR |
| 32 | G | V | S | E | M | T | I | Ec DeoR |
| 33 | G | V | S | P | G | T | I | Ec RpoD |
| 76 | G | A | G | I | A | T | I | Ec TrpR |
| 178 | G | C | S | R | E | T | V | Ec CAP |
| 205 | C | L | S | P | S | R | L | Ec AraC |
| 210 | C | L | S | P | S | R | L | St AraC |

size alphabet and very long sequences, the sequential pattern analysis can be applied effectively to the problem of motif finding between bioinformatics sequence and classification protein sequences based on their structures. Before the mining phase in almost all of sequential pattern mining algorithms, the main task is checking if one sequence is a subsequence of another sequence in the database to support the finding the frequent sequence. In this part, we will describe about subsequence mapping method, which is the preprocessing step for sequential pattern mining, for sequences in the CATH bioinformatics database.

The CATH database is a hierarchical domain classification of protein structures in the Protein Data Bank (PDB). In this database, protein structures are classified using a combination of automated and manual procedures. There are four major levels in this hierarchy: Class, Architecture, Topology (fold family) and Homologous superfamily. Each level is assigned a unique numeric label ('CATH number'). We use sequences in H-level, which groups together protein domains which are thought to share a common ancestor and can therefore be described as homologous. CATH sequence database which is considered as the data-sequence was built as form (CATH code, ProteinID, Protein sequence)

In the H-level which is corresponding to CATH code, there are many protein sequences in FASTA form with their protein IDs (PDB CATHcode). Each protein families are often characterized by one or more motifs ? which are regions or portions of protein sequences that has a specific structure and are functionally significant. Based on each protein sequence and motifs with the same length at the difference positions in protein sequence as in the following example table [Gira Narasimhan, 2002], our preprocessing step is finding the set of maximal pattern

### 2) Method

As definition in the section 2, the frequent pattern is called maximal if it is not in any other significant pattern (Maximal Pattern - MP). With the input is the set of aligned motif in specified length and the user min-support, using the following FindMP algorithm to find the set of the maximal pattern. The results are in Table 4.

```
FindMP(aligned motifs in CATH protein sequence, min-sup)
 L1 = {1 _ frequent sequence pattern}
 For i = 2 to length-of-motif
  For every pairs pattens p, q in Li-1
  If p, q share the first (i-2) items in common
        Insert f = p ∪ q into list Ei
  For every pattern f in Ei
```

Table 4. Result of *FindMP* algorithm.

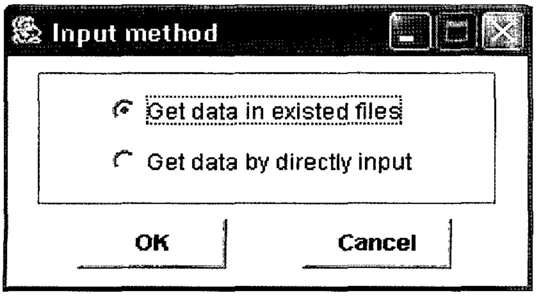
| Min-support 3 | | |
|---|---|---|
| Pattern length | MPs | Support |
| 2 | {S3, P4} | 3 |
| 2 | {S3, S5} | 3 |
| 3 | {G1, T6, I7} | 3 |
| 3 | {G1, S3, T6} | 3 |
| 3 | {G1, V2, S3} | 3 |

```
If support(f) > min-sup
        Insert f into Li
        Remove all subsets of f from Li-1
Return L = ∪i Li
```

From the protein sequences in corresponded CATH code and the set of maximal pattern which is the result of preprocessing frequent mining process, we will map them to create the sequence database as form (CATH code, MPs). For each protein in each class, we will check the list of MP in order. In one class, there are many protein sequences and each sequence can contain many different maximal patterns.

The subsequence mapping is finding what maximal patterns in each class to build the data sequence database for sequential pattern discovery. Each row in the mapping result is the transaction which has transaction ID as CATH class code and itemsets as the list of maximal pattern contained in that class. Based on that data sequence database, after the first step - subsequence mapping between protein class and maximal, we use ApioriAll algorithm to find all sequential patterns in it based on the user support threshold.

```
AprioriAll ((CATH code, MPs) database, min-sup)
L1 = {1 - frequent sequence pattern};
int k = 1;
while ($L_{k-1} \neq \emptyset$) {
$C_k$ = candGen($L_{k-1}$);
$C_k$ = candPrun($C_k$);
$L_k$ = supBasedPrun($C_k$);
k ← k + 1
}
Return C = $\cup_k C_k$
```



● Get data in existed files

○ Get data by directly input

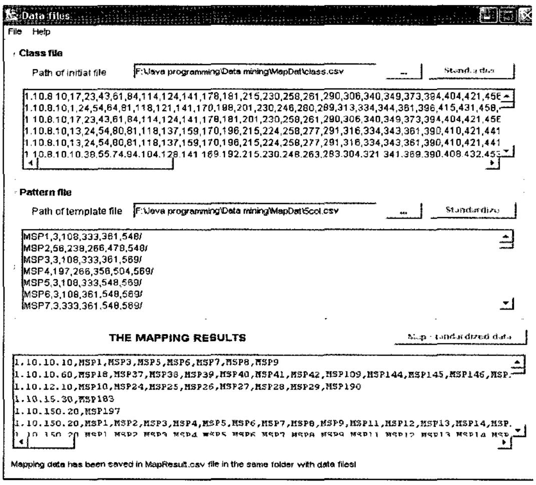
OK Cancel

Fig. 1. The optional user interface for input way.

## 4. Implementation

For building this method with the user interface, we used WindowXP as operating system and Java (version 1.5+) which supports for network and user interface programming as programming language. To simple the raw sequence data which has the form as amino acid chain, we normalize them by making index of amino acid. When mapping, all of the protein sequence and maximal pattern have number chain form.

With the first simple user interface as Fig. 1, user can choose one of two ways to input data: choose the preprocessed file data or input data directly for mapping. The raw data in number chain are saved in the file *.csv (Comma Separated Values). When choose one of these files to process, it has some unwanted elements which make the mapping difficult such as blank, redundant comma...Because of this



Class file

Path of initial file: F:\Java programming\Data mining\MapDat\class.csv

```
1.10.8 10,17,23,43,61,84,114,124,141,178,181,215,230,258,261,290,306,340,349,373,394,404,421,456
1.10.8.10,1,24,54,64,81,118,121,141,170,198,201,230,246,280,289,313,334,344,361,396,415,431,458,
1 10.8.10,17,23,43,61,84,114,124,141,178,181,201,230,258,261,290,306,340,349,373,394,404,421,456
1.10.8.10,13,24,54,80,81,118,137,159,170,196,215,224,258,277,291,316,334,343,361,390,410,421,441
1.10.8.10,13,24,54,80,81,118,137,159,170,196,215,224,258,277,291,316,334,343,361,390,410,421,441
1 10.8.10.10.38.55.74.94.104.128.141 169.192.215.230.248.263.283.304.321 341.369.390.408.432.453
```

Pattern file

Path of template file: F:\Java programming\Data mining\MapDat\Scol.csv

```
MSP1,3,108,333,361,548/
MSP2,56,238,266,478,548/
MSP3,3,108,333,361,569/
MSP4,197,266,356,504,569/
MSP5,3,108,333,548,569/
MSP6,3,108,361,548,569/
MSP7,3,333,361,548,569/
```

THE MAPPING RESULTS

```
1.10.10.10,MSP1,MSP3,MSP5,MSP6,MSP7,MSP8,MSP9
1.10.10.60,MSP18,MSP37,MSP38,MSP39,MSP40,MSP41,MSP42,MSP109,MSP144,MSP145,MSP146,MSP.
1.10.12.10,MSP10,MSP24,MSP25,MSP26,MSP27,MSP28,MSP29,MSP190
1.10.15.30,MSP183
1.10.150.20,MSP197
1.10.150.20,MSP1,MSP2,MSP3,MSP4,MSP5,MSP6,MSP7,MSP8,MSP9,MSP11,MSP12,MSP13,MSP14,MSP.
1.10.150.20,MSP1,MSP2,MSP3,MSP4,MSP5,MSP6,MSP7,MSP8,MSP9,MSP11,MSP12,MSP13,MSP14,MSP.
```

Mapping data has been saved in MapResult.csv file in the same folder with data files!

Fig. 2. The main user interface.

reason, almost all of these file have to standardize for deleting redundant symbols. After standardizing them, we begin to do subsequence mapping method to retrieve data – sequence as form (CATH, MPs). This method with above manipulations is implemented in the simple user interface in the Fig. 2.

## 5. Conclusion and Future Works

In this paper, we have presented the subsequence mapping method which is applied in bioinformatics sequences as the preprocessing steps for sequential pattern discovery. In protein sequence, some regions which are better conserved than others during evolution are called motif. The protein families are characterized by one or more motifs and their classification are represented in CATH database. This database with the H-level groups together protein domains which are thought to share a common ancestor and can therefore be described as homologous. Associating knowledge about motif and data mining techniques, namely sequential pattern mining algorithm, we can statistic the relationship between the class of protein family and the characteristic motif of protein sequence.

We are actively working in applying the more effective sequential pattern mining algorithm after mapping bioinformatics sequence. Because the imposition of a gap restriction is critical for the problem of motif finding in protein sequences, the improvement of this work is using the pattern-growth algorithm while apriori-based methods are inapplicable in such problem.

## Acknowledgements

We would like to thank the Regional research Centers Program of Ministry of Education and Human Resources Development in Korea for supporting this research.

## References

R. Agrawal and R. Srikant, 1995. *Mining sequential Patterns*, In Proc. of Intl. Conf. on Data Engineering, p. 3-14, Taipei, Taiwan.

Srikant, R. and Agrawal, R., 1996 *Mining sequential Patterns: Generalized and Performance improvements.* Proceedings of International Conference on Extending Database Technology, p. 3-17.

Zaki M, 1998. *Efficient Enumeartion of Frequent Sequences.* ACM Conf. on Information Knowledge Management, p. 68-75.

J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M. -C. Hsu, 2000. *FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining.* Proc. 2000 ACM SIGKDD Int'l Conf. Knowledge Discovery in Databases (KDD '00) pp. 355-359.

J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. -C. Hsu, 2001. *PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth.* Proc. 2001 Int'l Conf. Data Eng. (ICDE '01), pp. 215-224.

Gira Narasimhan, Changsong Bu, Yuan Gao, Xuning Wang, Ning Xu, and Kalai Mathee, 2002. *Mining Protein Sequences for Motifs.* Journal of Computational Biology, 9(5): 707-720.

Margaret H. Dunham, Southern Methodist University. *Data mining-Introductory and Advanced Topics.*

Jason T. L. Wang, Mohammed J. Zaki, Hannu T. T. Toivonen, Dennis Shasha, *Data Mining in Bioinformatics.*