Exploring Lattice Structures in Mining Multi-Domain Sequential Patterns

Zhung-Xun Liao and Wen-Chih Peng
Department of Computer Science
National Chiao Tung University
Hsinchu, Taiwan, ROC
E-mail:{wcpeng, g9113}@cs.nctu.edu.tw

***Abstract*— Since sequential patterns may exist in multiple sequence databases, we propose algorithm PropagatedMine$^+$ to efficiently discover multi-domain sequential patterns. Prior works have shown that algorithm PropagatedMine outperforms other methods. In this paper, by exploring lattice structures, we develop algorithm PropagatedMine$^+$ for propagating. Note that the lattice structure provides some guidelines when mining sequential patterns in other domain databases. Thus, exploiting the lattice structure devised could further reduce the number of candidates patterns, thereby improving the performance of mining sequential patterns across multiple domain sequence databases. A comprehensive performance study is conducted and experimental results show the scalability and the efficiency of algorithm PropagatedMine$^+$.**

I. Introduction

Existing sequential pattern mining algorithms [1][2][3] only discover the sequential behavior (e.g., buying behavior) in one domain, which are not sufficient. One would like to discover sequential patterns across multiple domains. Such a sequential pattern across multiple domain sequence databases is referred to a multi-domain sequential pattern [4]. Consider a mobile computing environment, where mobile users can access three services (i.e., location tracking service, data searching service, and credit payment service) via mobile devices and each service is referred to one domain in this paper. Given a log of movements of a user from the location tracking service, one would mine user moving patterns referred to those areas in which the user frequently travels. However, in order to reflect the behavior of a user in such environment, one would like to find more complex sequential patterns cross multiple domains, instead of only the moving patterns. An example of a multi-domain sequential pattern is shown in Table I, where a user stays at area $\{A\}$, searches data items $\{1, 2\}$, and buys goods $\{\alpha, \beta\}$; then moves to area $\{B, C\}$, searches data $\{3, 4, 5\}$, and buys goods $\{\gamma\}$; and finally moves to area $\{D\}$, searches data $\{6, 7\}$, and buys goods $\{\theta, \delta\}$. Such a sequential pattern consists of sequences cross multiple domains and provides more information to analyze user behaviors.

II. Challenges

Table II depicts multiple domain sequence databases and each domain sequence database are stored individually. Notice that time instance sequences represent the occurred time slots for the corresponding sequences in each domain sequence database. The prior work [4] has formulated the problem of mining sequential patterns across multiple domain sequence databases. Therefore, the main challenge is to discover sequential patterns across multiple sequence databases without joining every domain sequence database via time instance sequences.

Location tracking	(A)	(B, C)	(D)
Search	(1, 2)	(3, 4, 5)	(6, 7)
Payment	(α, β)	(γ)	(θ, δ)

TABLE I
An example of a multi-domain sequential pattern.

III. Algorithm PropagatedMine$^+$

The prior work in [4] developed algorithm PropagatedMine to propagate sequential patterns mined to other domain sequence databases. Though algorithm PropagatedMine is able to outperform other methods, the cost of propagating sequential patterns could be further reduced. Thus, in this paper, we propose algorithms PropagatedMine$^+$. The same to the work in [4], algorithm PropagatedMine$^+$ consists of two phases: the mining phase and the deriving phase. In the mining phase, algorithm PropagatedMine$^+$ utilizes existing sequential pattern mining algorithms to discover sequential patterns in a starting domain and then propagates time instance sets of only 1-sequences (referred to as atomic patterns) to next domains. However, the sequential patterns in the starting domain are represented as the lattice graph structure to facilitate the generation of candidates and provide guidelines for mining multi-domain sequential patterns. For example, assume that the starting domain is set to D_1 in Table II. Those sequential patterns mined are represented as a lattice structure shown in Fig. 1, where each node represents a sequential pattern and the linkages of nodes (standing for *intra-domain links*) represent itemset relationships. Furthermore, those nodes having the same number of elements are further arranged level-by-level. Explicitly, it can be seen in Fig. 1 for the nodes with their number of elements is 1, these nodes are put level-by-level in increasing order of length of sequences.

Domain database D_1		
Id	Time instance sequences	Sequences
s_1	<(T_1)(T_2)(T_3)(T_4)>	<(a)(b,c)(b,c,d)(e)>
s_2	<(T_5)(T_7)(T_8)>	<(a,b)(b,c)(c,e)>
s_3	<(T_{10})(T_{12})(T_{13})>	<(a,e)(h)(g,j)>
s_4	<(T_{21})(T_{22})(T_{23})(T_{24})>	<(a,b,f)(d)(b,c)(e,f)>

Domain database D_2		
Id	Time instance sequences	Sequences
l_1	<(T_{21})(T_{22})(T_{23})(T_{24})>	<(1,2,5)(7)(2,3)(4,5,6)>
l_2	<(T_{10})(T_{12})(T_{13})>	<(1,6)(5)(9,10)>
l_3	<(T_5)(T_7)(T_8)>	<(1,3)(2,4)(8)>
l_4	<(T_1)(T_2)(T_3)(T_4)>	<(1,2)(2,3)(6)(4,5)>

TABLE II

AN EXAMPLE OF MULTIPLE DOMAIN SEQUENCE DATABASES

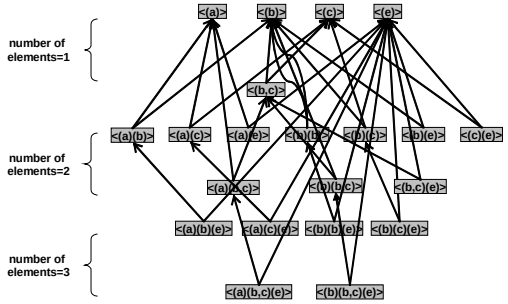

Fig. 1. A lattice structure for sequential patterns in a starting domain (i.e., D_1 in Table 2).

Definition 1 (Propagated table): Let M be a multi-domain sequential pattern across k domain sequence databases with $TIS(M) = \{< TS_1 : l_1 >, < TS_2 : l_2 >, ..., < TS_f : l_f >\}$, where TS_i is a time instance sequence and l_i is the corresponding integer list. Assume that domain $D_t = \{s_1, s_2, ..., s_m\}$, where $s_i =< X_1^i, X_2^i, ..., X_{e(t_i)}^i >$ and each sequence s_i has the corresponding time instance sequence, denoted as TS_{s_i}. When propagating time instance sets of M to domain D_t, we could have a propagated table defined as $D_t||_M = \{X_{l_j}^i | \exists TS_{s_i}$ and $TS_j \ni (TS_{s_i} = TS_j)\}$.

For example, in Table II, by propagating $TIS(< (b) >)$ to sequence database D_2, we could have the propagated table $D_2||_{<(b)>}$ shown in Table III. After obtaining propagated tables, we could mine frequent itemsets by association rule mining algorithms. Then, those frequent itemsets could be combined by the corresponding patterns propagated to generate multi-domain sequence patterns. From the above example, given a minimum support 3, we can easily obtain $\begin{bmatrix} (b) \\ (2) \end{bmatrix}$ and $\begin{bmatrix} (b) \\ (3) \end{bmatrix}$ as multi-domain sequence patterns across 2 domain sequence databases, where (2) and (3) are the frequent items of $D_2||_{<(b)>}$.

The detailed steps for deriving phase are described as follows:

Step 1: Derive through propagated table:

In Table II, we first derive atomic patterns in sequence database D_2. Specifically, in Fig. 2, time instance sets of atomic patterns in sequence database D_1 (i.e., the top-level nodes) are propagated to sequence database D_2 II. From the propagated table of each atomic pattern, atomic patterns are easily obtained. For each atomic pattern in D_1, there is a *inter-domain link* representing that these two patterns are able to form multi-domain sequential patterns. Consequently, we have $\begin{bmatrix} (a) \\ (1) \end{bmatrix} \begin{bmatrix} (b) \\ (2) \end{bmatrix} \begin{bmatrix} (b) \\ (3) \end{bmatrix} \begin{bmatrix} (c) \\ (2) \end{bmatrix}$ in the above example.

Time instance sequences	Items
<(T_1)(T_2)(T_3)(T_4)>	(2,3)
<(T_1)(T_2)(T_3)(T_4)>	(6)
<(T_5)(T_7)(T_8)>	(1,3)
<(T_5)(T_7)(T_8)>	(2,4)
<(T_{21})(T_{22})(T_{23})(T_{24})>	(1,2,5)
<(T_{21})(T_{22})(T_{23})(T_{24})>	(2,3)

TABLE III

AN EXAMPLE OF PROPAGATED TABLE $D_2||_{<(b)>}$.

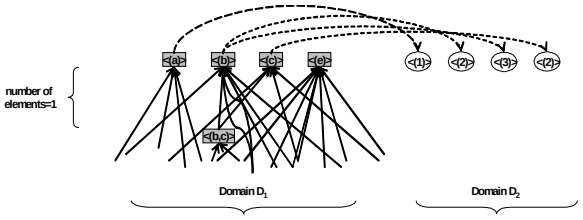

Fig. 2. An example of generating atomic patterns in domain D_2.

Step 2: Derive through union operation:

In this step, we will derive multi-domain sequential patterns with their number of elements to be one. For example, let $Q =< (b, c) >$, a sequential pattern with $e(Q) = 1$ in domain D_1 of Table II. Through the intra-domain links, we can find atomic patterns that are components of Q (i.e., $< (b) >$ and $< (c) >$). In Fig. 3 following inter-domain links of $< (b) >$ and $< (c) >$, we could obtain the atomic patterns in domain D_2 (i.e., $< (2) >$ and $< (3) >$). Consequently, two possible unions of P are generated (i.e., $\begin{bmatrix} (b) \\ (2) \end{bmatrix} \cup \begin{bmatrix} (c) \\ (2) \end{bmatrix} = \begin{bmatrix} (b,c) \\ (2) \end{bmatrix}$ and $\begin{bmatrix} (b) \\ (3) \end{bmatrix} \cup \begin{bmatrix} (c) \\ (2) \end{bmatrix} = \begin{bmatrix} (b,c) \\ (2,3) \end{bmatrix}$). Once we have the possible candidate multi-domain sequence patterns, support values of these patterns are examined by checking their time instance sets. Given a minimum support 3, since the support values of $\begin{bmatrix} (b,c) \\ (2) \end{bmatrix}$ and $\begin{bmatrix} (b,c) \\ (2,3) \end{bmatrix}$ are 3 and 2, respectively, $\begin{bmatrix} (b,c) \\ (2) \end{bmatrix}$ is a frequent multi-domain sequence. Thus, the lattice structure in domain D_2 contains $< (2) >$ and inter-domain links are built between lattice structures in

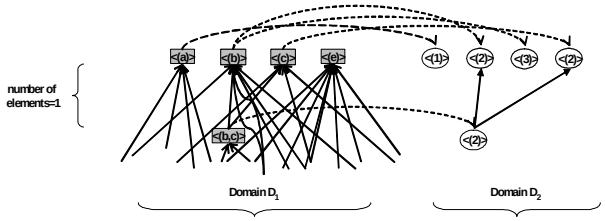

Fig. 3. An example of generating sequential patterns whose number of elements is 1 in domain D_2.

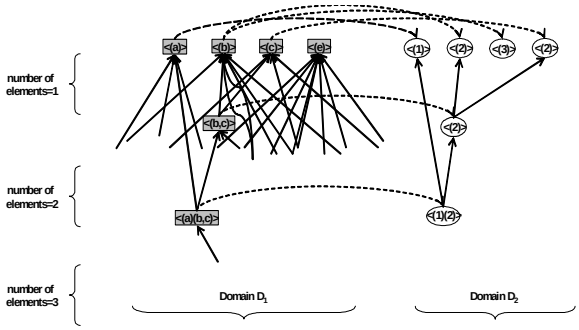

Fig. 4. An example of generating sequential patterns with their number of elements larger than 1 in domain D_2.

domain D_1 and that in domain D_2.

Step 3: Derive through concatenate operation:

Before describing this step, we first define the concatenate operation.

Definition 2 (concatenate operation of TIS): Let M and N be two multi-domain sequence. $TIS(M) = \{< S_1 : l_{11}, l_{12}, ..., l_{1e(M)} >, < S_2 : l_{21}, l_{22}, ..., l_{2e(M)} >, ..., < S_m : l_{m1}, l_{m2}, ..., l_{me(M)} >\}$ and $TIS(N) = \{< T_1 : k_{11}, k_{12}, ..., k_{1e(N)} >, < T_2 : k_{21}, k_{22}, ..., k_{2e(N)} >, ..., < T_n : k_{n1}, k_{n2}, ..., k_{ne(N)} >\}$. The concatenation of $TIS(M)$ and $TIS(N)$ is denoted as $TIS(M) \cap_< TIS(N) = \{< S_i : l_{i1}, l_{i2}, ..., l_{ie(M)}, k_{j1}, k_{j2}, ..., k_{je(N)} >\}$ such that $S_i = T_j$ and $l_{ie(M)} < k_{j1}$. In other words, $TIS(M) \cap_< TIS(N)$ is the time instance set of the multi-domain sequence $< M, N >$.

For example, given $M = \begin{bmatrix} (a) \\ (1) \end{bmatrix}$, $N = \begin{bmatrix} (b,c) \\ (2) \end{bmatrix}$ and the multi-domain sequence database in Table II with a minimum support as 3, it can be verified that $TIS(M) = \{< (T_1)(T_2)(T_3)(T_4) : 1 >, < (T_5)(T_7)(T_8) : 1 >, < (T_{10})(T_{12})(T_{13}) : 1 >, < (T_{21})(T_{22})(T_{23})(T_{24}) : 1 >\}$, $TIS(N) = \{< (T_1)(T_2)(T_3)(T_4) : 2 >, < (T_5)(T_7)(T_8) : 2 >, < (T_{21})(T_{22})(T_{23})(T_{24}) : 3 >\}$, and $TIS(M) \cap_< TIS(N) = \{< (T_1)(T_2)(T_3)(T_4) : 1, 2 >, < (T_5)(T_7)(T_8) : 1, 2 >, < (T_{21})(T_{22})(T_{23})(T_{24}) : 1, 3 >\}$. From $TIS(M) \cap_< TIS(N)$, we could further merge these sequential patterns into $\begin{bmatrix} (a) & (b,c) \\ (1) & (2) \end{bmatrix}$.

The multi-domain sequential patterns with the number of elements larger than 1 will be derived in this step. Consider an example pattern $P =< (a)(b,c) >$ in Fig. 4. By intra-domain links and inter-domain links, we have $\begin{bmatrix} (a) \\ (1) \end{bmatrix} \cap_< \begin{bmatrix} (b,c) \\ (2) \end{bmatrix}$. Therefore, $P' = \begin{bmatrix} (a)(b,c) \\ (1)(2) \end{bmatrix}$ is generated.

Algorithm PropagatedMine$^+$ iteratively repeats the above three steps until all domain sequence databases are propagated.

Correctness of PropagatedMine$^+$: The correctness of step 1 is omitted. Interested readers could refer to the prior work in [4]. In this paper, we only prove the correctness of step 2 and step 3. Let P be a k-domain sequential pattern and P' be a $(k+1)$-domain sequential pattern derived from P, where $e(P') = e(P) = 1$, $|P'| \geqslant |P| > 1$, and the corresponding frequent itemset of P and P' is Z. In other words, $P' = \begin{bmatrix} P \\ Z \end{bmatrix}$. Given an arbitrary division X and Y of P, such that $X \cup Y = P$. Clearly, there are intra-domain links from P to X and Y. In addition, there are inter-domain links from X and Y to Z', where $Z' \in P(Z)$ the power set of Z and $Z' \neq \emptyset$, due to the anti-monotone property (i.e., if P' is frequent, all multi-domain sequences contained by P' must be frequent, too). Hence, both $\begin{bmatrix} X \\ Z' \end{bmatrix}$ and $\begin{bmatrix} Y \\ Z' \end{bmatrix}$ are frequent. Therefore, the lattice structures could derive every pairs of P and P'. Following the above operations, the correctness of step 3 is also applied.

IV. Experimental Results

We first investigate the performance of algorithms PropagatedMine and PropagatedMine$^+$ with the value of the minimum support varied from 0.5% to 5%. The execution time of these two algorithms is shown in Fig. 5. With the smaller minimum support, the number of sequential patterns will be larger, thereby increasing the execution time of both algorithms. Since algorithm PropagatedMine needs to propagate more multi-domain sequential patterns, the execution time of algorithm PropagatedMine is larger than that of algorithm PropagatedMine$^+$.

Next, we conduct experiments with the number of domain varied from 2 to 5 and the minimum support is set to 0.3%. The performance is shown in Fig. 6. Clearly, when the number of domains increases, the execution time of both algorithms PropagatedMine and PropagatedMine$^+$ increases. It is expected that with a larger number of domains, algorithm PropagatedMine performs worse than algorithm PropagatedMine$^+$ since more propagated tables are generated.

The experiments of varying the number of sequences is now evaluated. The numbers of sequences are set to 1000, 2000, 3000, 4000, 5000 and 6000, respectively. The setting of the minimum support is 1%. As can be seen in Fig. 7, the execution time of both algorithms increases with the number of sequences. Furthermore, algorithm PropagatedMine$^+$ outperforms algorithm PropagatedMine due to the same reason that algorithm PropagatedMine needs to process every multi-domain sequential patterns. As the number of sequences increases, the total number of multi-domain sequential patterns becomes larger.

Algorithm PropagatedMine$^+$:
Input: Multi-domain sequence database with n domains, $D_1, D_2, ..., D_n$, and the minimum support δ.
Output: Multi-domain sequential patterns with n domains.
Begin
Apply sequential pattern mining on D_1.
Let SP_1 be the set of sequential patterns mined in D_1.
For each domain D_i, $i = 2, 3, ..., n$
For each $P \in SP_{i-1}$
If $|P| = 1$ **Then**
Construct Propagation Table $D_i||_P$.
Find frequent items in $D_i||_P$ with minimum support δ.
Let FI be the set of frequent items in $D_i||_P$.
For each $q \in FI$
Append $\begin{bmatrix} P \\ q \end{bmatrix}$ to SP_i.
Let $TIS(\begin{bmatrix} P \\ q \end{bmatrix}) = TIS(P) \cap TIS(q)$.
If $e(P) = 1$ **Then**
Compose $\begin{bmatrix} P \\ q \end{bmatrix}$ with $e(\begin{bmatrix} P \\ q \end{bmatrix}) = 1$.
If $Support(\begin{bmatrix} P \\ q \end{bmatrix}) \geqslant \delta$ **Then**
append $\begin{bmatrix} P \\ q \end{bmatrix}$ to SP_i
If $e(P) > 1$ **Then**
Compose $\begin{bmatrix} P \\ q \end{bmatrix}$ with $e(\begin{bmatrix} P \\ q \end{bmatrix}) > 1$.
If $Support(\begin{bmatrix} P \\ q \end{bmatrix}) \geqslant \delta$ **Then**
append $\begin{bmatrix} P \\ q \end{bmatrix}$ to SP_i
Output=SP_n.
End

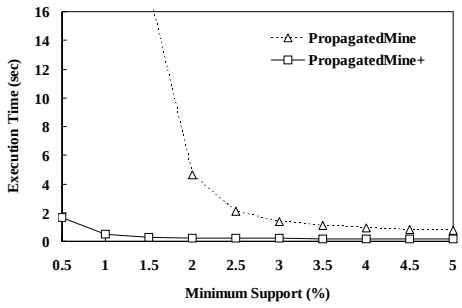

Fig. 5. The execution time of algorithms with various minimum support values.

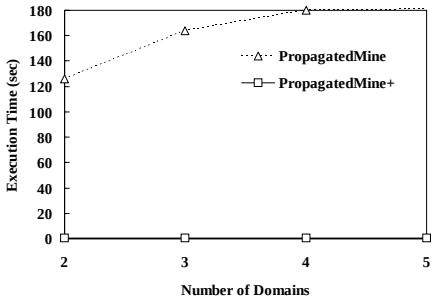

Fig. 6. The performance of algorithms with the number of domain varied.

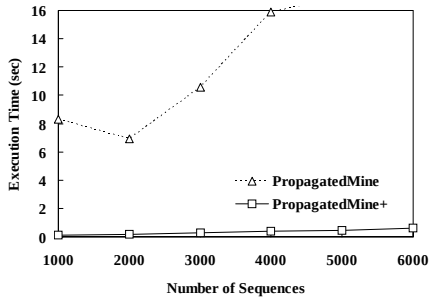

Fig. 7. The performance of algorithms with the number of sequences varied.

V. Conclusions

In this paper, we proposed algorithm PropagatedMine$^+$ for mining sequential patterns across multiple domain sequence databases. Algorithm PropagatedMine$^+$ first mines sequential patterns in a starting domain sequence database, and then uses a lattice structure to store these sequential patterns. In light of the lattice structure, algorithm PropagatedMine$^+$ is able to propagate time instance sets of only atomic patterns to next domains for mining sequential patterns in a level-by-level manner. A comprehensive performance study was conducted. Experimental results show that by only propagating time instance sets of atomic patterns to other domains, algorithm PropagatedMine$^+$ outperforms algorithm PropagatedMine.

References

[1] R. Agrawal and R. Srikant, "Mining sequential patterns." in *Proceedings of the 1995 IEEE International Confernece on Data Engineering*, 1995, pp. 3–14.

[2] H. Cheng, X. Yan, and J. Han, "Incspan: Incremental mining of sequential patterns in large database." in *Proceedings of the 2004 ACM SIGKDD International Confernece on Data Mining*, 2004, pp. 527–532.

[3] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, "Prefixspan: Mining sequential patterns by prefix-projected growth." in *Proceedings of the 2001 IEEE International Confernece on Data Engineering*, 2001, pp. 215–224.

[4] Z.-X. Liao, W.-C. Peng, and X.-Y. Hu, "Mining multi-domain sequential patterns," in *Workshop on Software Engineering, Databases, and Knowledge Discovery, International Computer Symposium*, 2006, pp. 334–339.