

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, g91113}@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 candidate 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⁺.

1. 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 across multiple domains, instead of only the moving patterns. An example of a multi-domain sequential pattern is shown in Table 1, where a user stays at area $\{A\}$, searches data items $\{1, 2\}$, and buys goods $\{\alpha, \beta\}$;

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

INFOSCALE 2007, June 6-8, Suzhou, China
Copyright © 2007 ICST 978-1-59593-757-5
DOI 10.4108/infoscale.2007.909

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

Table 1: An example of a multi-domain sequential pattern.

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 across multiple domains and provides more information to analyze user behaviors.

2. CHALLENGES

Table 2 depicts multiple domain sequence databases and each domain sequence database is 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.

3. 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 algorithm 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 2. Those sequential patterns mined are represented as a lattice structure shown in Figure 1, where each node

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

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

Table 2: An example of multiple domain sequence databases

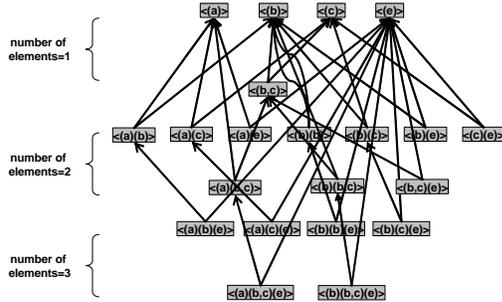


Figure 1: A lattice structure for sequential patterns in a starting domain (i.e., D_1 in Table 2).

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 Figure 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.

Definition 1 (Propagated table): Let M be a multi-domain sequential pattern across k domain sequence databases with $TIS(M) = \{ \langle TS_1 : l_1 \rangle, \langle TS_2 : l_2 \rangle, \dots, \langle TS_f : l_f \rangle \}$, where TS_i is a time instance sequence and l_i is the corresponding integer list. Assume that domain $D_t = \{s_1, s_2, \dots, s_m\}$, where $s_i = \langle X_1^i, X_2^i, \dots, X_{e(t)}^i \rangle$ 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 $D_t||_M = \{X_j^i | \exists TS_{s_i} \text{ and } TS_j \ni (TS_{s_i} = TS_j)\}$.

For example, in Table 2, by propagating $TIS(\langle (b) \rangle)$ to sequence database D_2 , we could have the propagated table $D_2||_{\langle (b) \rangle}$ shown in Table 3. 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 sequential 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 sequential patterns across 2 domain sequence databases, where (2) and (3) are the frequent items of $D_2||_{\langle (b) \rangle}$.

The detailed steps for deriving phase are described as follows:

Step 1: Derive through propagated table:

In Table 2, we first derive atomic patterns in sequence database D_2 . Specifically, in Figure 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 . From the

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

Table 3: An example of propagated table $D_2||_{\langle (b) \rangle}$.

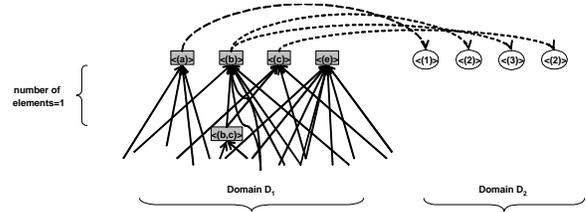


Figure 2: An example of generating atomic patterns in domain D_2 .

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 $\begin{bmatrix} (a) \\ (1) \end{bmatrix}$, $\begin{bmatrix} (b) \\ (2) \end{bmatrix}$, $\begin{bmatrix} (b) \\ (3) \end{bmatrix}$ and $\begin{bmatrix} (c) \\ (2) \end{bmatrix}$ in the above example.

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 $Q = \langle (b, c) \rangle$, a sequential pattern with $e(Q) = 1$ in domain D_1 of Table 2. Through the intra-domain links, we can find atomic patterns that are components of Q (i.e., $\langle (b) \rangle$ and $\langle (c) \rangle$). In Figure 3 following inter-domain links of $\langle (b) \rangle$ and $\langle (c) \rangle$, we could obtain the atomic patterns in domain D_2 (i.e., $\langle (2) \rangle$ and $\langle (3) \rangle$). 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 sequential 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

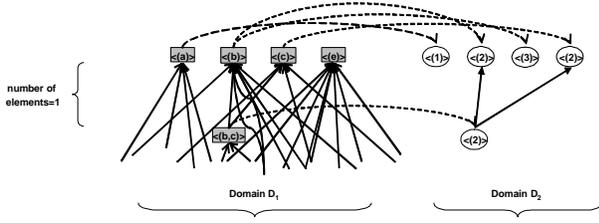


Figure 3: An example of generating sequential patterns whose number of elements is 1 in domain D_2 .

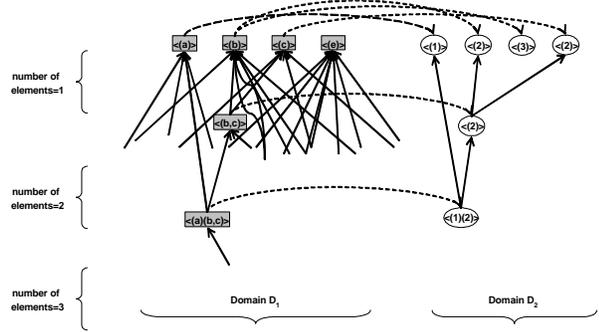


Figure 4: An example of generating sequential patterns with their number of elements larger than 1 in domain D_2 .

$\langle (2) \rangle$ and inter-domain links are built between lattice structures in 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 sequences. $TIS(M) = \{ \langle S_1 : l_{11}, l_{12}, \dots, l_{1e(M)} \rangle, \langle S_2 : l_{21}, l_{22}, \dots, l_{2e(M)} \rangle, \dots, \langle S_m : l_{m1}, l_{m2}, \dots, l_{me(M)} \rangle \}$ and $TIS(N) = \{ \langle T_1 : k_{11}, k_{12}, \dots, k_{1e(N)} \rangle, \langle T_2 : k_{21}, k_{22}, \dots, k_{2e(N)} \rangle, \dots, \langle T_n : k_{n1}, k_{n2}, \dots, k_{ne(N)} \rangle \}$. The concatenation of $TIS(M)$ and $TIS(N)$ is denoted as $TIS(M) \cap_{\langle} TIS(N) = \{ \langle S_i : l_{i1}, l_{i2}, \dots, l_{ie(M)}, k_{j1}, k_{j2}, \dots, k_{je(N)} \rangle \}$ such that $S_i = T_j$ and $l_{ie(M)} < k_{j1}$. In other words, $TIS(M) \cap_{\langle} TIS(N)$ is the time instance set of the multi-domain sequence $\langle M, N \rangle$.

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 2 with a minimum support as 3, it can be verified that $TIS(M) = \{ \langle (T_1)(T_2)(T_3)(T_4) : 1 \rangle, \langle (T_5)(T_7)(T_8) : 1 \rangle, \langle (T_{10})(T_{12})(T_{13}) : 1 \rangle, \langle (T_{21})(T_{22})(T_{23})(T_{24}) : 1 \rangle \}$, $TIS(N) = \{ \langle (T_1)(T_2)(T_3)(T_4) : 2 \rangle, \langle (T_5)(T_7)(T_8) : 2 \rangle, \langle (T_{21})(T_{22})(T_{23})(T_{24}) : 3 \rangle \}$, and $TIS(M) \cap_{\langle} TIS(N) = \{ \langle (T_1)(T_2)(T_3)(T_4) : 1, 2 \rangle, \langle (T_5)(T_7)(T_8) : 1, 2 \rangle, \langle (T_{21})(T_{22})(T_{23})(T_{24}) : 1, 3 \rangle \}$. From $TIS(M) \cap_{\langle} TIS(N)$, 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 $P = \langle (a)(b, c) \rangle$ in Figure 4. By intra-domain links and inter-domain links, we have $\begin{bmatrix} (a) \\ (1) \end{bmatrix} \cap_{\langle}$

$\begin{bmatrix} (b, c) \\ (2) \end{bmatrix}$. 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 $e(P') = e(P) = 1$, $|P'| \geq |P| > 1$, 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.

Algorithm PropagatedMine⁺:

Input: Multi-domain sequence database with n domains, D_1, D_2, \dots, 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 $D_i, i = 2, 3, \dots, n$

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 $\min_sup \delta$.

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 $Support(\begin{bmatrix} P \\ q \end{bmatrix}) \geq \delta$ **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 $Support(\begin{bmatrix} P \\ q \end{bmatrix}) \geq \delta$ **Then**

append $\begin{bmatrix} P \\ q \end{bmatrix}$ to SP_i

Output= SP_n .

End

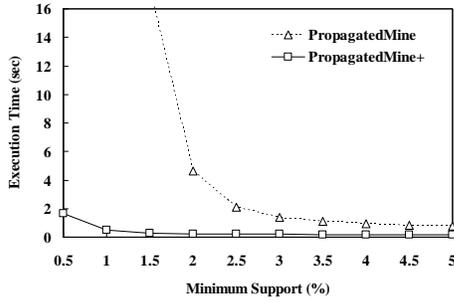


Figure 5: The execution time of algorithms with various minimum support values.

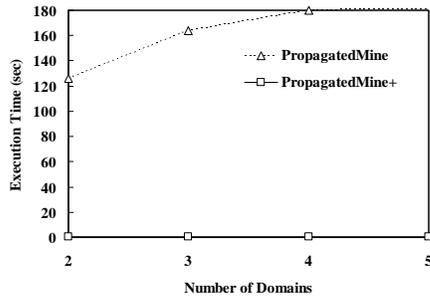


Figure 6: The performance of algorithms with the number of domain varied.

4. EXPERIMENTAL RESULTS

We first investigate the performance of algorithms PropagatedMine and PropagatedMine⁺ with the value of minimum support varied from 0.5% to 5%. The execution time of these two algorithms is shown in Figure 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 domains varied from 2 to 5 and the minimum support is set to 0.3%. The performance is shown in Figure 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 Figure 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

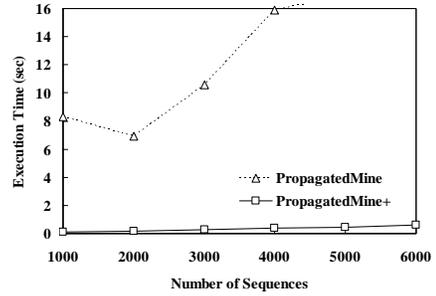


Figure 7: The performance of algorithms with the number of sequences varied.

multi-domain sequential patterns. As the number of sequences increases, the total number of multi-domain sequential patterns becomes larger.

5. 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.

6. REFERENCES

- [1] R. Agrawal and R. Srikant, "Mining sequential patterns." in *Proceedings of the 1995 IEEE International Conference on Data Engineering*, 1995, pp. 3–14.
- [2] H. Cheng, X. Yan, and J. Han, "Incsan: Incremental mining of sequential patterns in large database." in *Proceedings of the 2004 ACM SIGKDD International Conference 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 Conference 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.