On Transversal Hypergraph Enumeration in Mining Sequential Patterns

Haoyuan Li, Anne Laurent, Maguelonne Teisseire

On Transversal Hypergraph Enumeration in Mining Sequential Patterns

Dong (Haoyuan) Li
École des Mines d'Alès
LGI2P, Parc Georges Besse
30035 Nîmes
France
Haoyuan.Li@ema.fr

Anne Laurent
LIRMM, CNRS
161 rue Ada
34392 Montpellier
France
laurent@lirmm.fr

Maguelonne Teisseire
LIRMM, CNRS
161 rue Ada
34392 Montpellier
France
teisseire@lirmm.fr

Abstract

The transversal hypergraph enumeration based algorithms can be efficient in mining frequent itemsets, however it is difficult to apply them to sequence mining problems. In this paper we first analyze the constraints of using transversal hypergraph enumeration in itemset mining, then propose the ordered pattern *model for representing and mining sequences with respect to these constraints. We show that the problem of mining sequential patterns can be transformed to the problem of mining frequent ordered patterns, and therefore we propose an application of the* Dualize and Advance *algorithm, which is transversal hypergraph enumeration based, in mining sequential patterns.*

1. Introduction

As one of the most concentrated topics in data mining research, mining frequent itemsets has received much attention. A lot of algorithms for this problem have been developed [1] since the first introduction of the *apriori* (also called *level-wise*) algorithm in mining association rules [1, 2, 9], where how to efficiently find all frequent itemsets is the major subtask.

The *apriori* (*level-wise*) algorithm considers the *specialization relation* (denoted by $\preceq$) between itemsets, so that the search of all frequent itemsets can be performed by walking up in the subset lattice of itemsets imposed by $\preceq$, one level at a time. [8] detailed this approach. However, when the most specific frequent itemsets appear at high levels in this search, the number of all frequent itemsets may be too large and the time of computing may not be acceptable.

[6] proposed the *Dualize and Advance* algorithm that only finds all most specific frequent itemsets using irredundant dualization done by transversal hypergraph enumeration, instead of finding all frequent itemsets. This approach is limited to itemset mining because the application of transversal hypergraph enumeration requires that the structure of patterns being discovered (e.g. itemset) satisfies the "representing as sets".

In this paper we are interested in porting transversal hypergraph enumeration based algorithms to sequential pattern mining. The rest of this paper is organized as follows. Section 2 introduces the transversal hypergraph enumeration on itemset mining. In Section 3 we analyze the constraints on applying transversal hypergraph enumeration. In the next Section 4 we formalize sequences with our proposition of the *ordered pattern* with respect to the above constraints. In Section 5 we show that with ordered patterns the *Dualize and Advance* algorithm can be used in mining sequential patterns. The final section is a short conclusion.

2. Transversal Hypergraph Enumeration on Itemset Mining

Given a data set $\mathbf{r}$ over relation R, we use the term *language*, denoted by $\mathcal{L}$, for expressing properties or defining subgroups of the data. $\mathcal{L}$ represents the structure of patterns being discovered in data mining and the computational task can therefore be considered as finding all sentences $\varphi \in \mathcal{L}$ that defines a sufficiently large subclass of $\mathbf{r}$. With the *specialization relation* $\preceq$ between all sentences of $\mathcal{L}$ in $\mathbf{r}$, considering a set $\mathcal{S} \subseteq \mathcal{L}$ such that $\mathcal{S}$ is closed downwards under the relation $\preceq$, the *positive border* $\mathcal{B}d^+(\mathcal{S})$ consists of the most specific sentences in $\mathcal{S}$ and the *negative border* $\mathcal{B}d^-(\mathcal{S})$ consists of the most general sentences that are not in $\mathcal{S}$ [8]. Figure 1 illustrates the notion of border.

In the problem of mining frequent itemsets addressed by a frequency threshold, the positive border consists of a set of the most specific frequent itemsets and the negative border consists the most general non-frequent itemsets. That is, all

[1] See the FIMI (Frequent Itemset Mining Implementations Repository) Web site for a collection of implementations. (http://fimi.cs.helsinki.fi/)

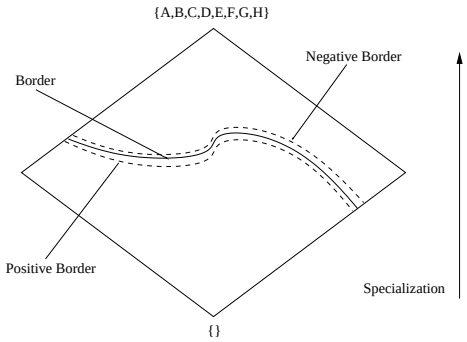

Figure 1. Border and specialization relation.

itemsets inside the border, from the vision of specialization, must be frequent but the ones outside the border cannot be frequent.

A collection $\mathcal{H}$ of subsets of R is a simple *hypergraph* [4], if no element of $\mathcal{H}$ is empty and if $X, Y \in \mathcal{H}$ and $X \subseteq Y$ imply $X = Y$. A hypergraph $\mathcal{H} = (V, E)$ consists of a finite collection E of sets over a finite set V. The elements of E are the *edges* of the hypergraph, and a *transversal* of $\mathcal{H}$ is a set $T \subseteq V$ that intersects all the edges of E. If no $T' \subset T$ is a transversal, we say that this transversal is *minimal*. The set $\mathcal{T}r(\mathcal{H})$ of all minimal transversals of a hypergraph $\mathcal{H}$ is called the *transversal hypergraph* of $\mathcal{H}$. [8] showed that if the language $\mathcal{L}$ satisfies the requirement of representing as sets, the negative border $\mathcal{B}d^-(\mathcal{S})$ can be computed by transversal hypergraph enumeration from the positive border, that is, $\mathcal{T}r(\overline{\mathcal{B}d^+(\mathcal{S})}) = \mathcal{B}d^-(\mathcal{S})$.

Definition 1 (Representing as Sets [6])**.** *Let $\mathcal{L}$ be the language, $\preceq$ a specialization relation, and $\mathcal{R}$ a set; denote by $\mathcal{P}(\mathcal{R})$ the powerset of R. A function $f : \mathcal{L} \to \mathcal{P}(\mathcal{R})$ is a representation of $\mathcal{L}$ (and $\preceq$) as sets, if f is one-to-one and surjective, f and its inverse are computable, and for all θ and φ we have $\varphi \preceq \theta$ if and only if $f(\varphi) \subseteq f(\theta)$. This transformation is called* representing as sets.

Based on the conclusions of [8], the *Dualize and Advance* algorithm first finds a most specific sentence, $\theta \in \mathcal{L}$ from an initial sentence φ, such that $\varphi \preceq \theta$. Once a set of most specific sentences is found, the algorithm computes the negative border of the sentences found by using transversal hypergraph enumeration, and restarts its upward search from this negative border. If progress can be made, the positive border can be made from the negative border and thus the approach is guaranteed to succeed. The time complexity of the *Dualize and Advance* algorithm depends on the complexity of transversal hypergraph enumeration. [5] presented an incremental algorithm for the transversal hypergraph computation with time complexity $T(I, i) = (I + i)^{O(\log(I+i))}$, thus the time complexity of the *Dualize and Advance* algorithm could be concluded as $t(|\mathcal{B}d^+| + |\mathcal{B}d^-|)$, where $t(n) = n^{\mathcal{O}(\log n)}$, while using at most $|\mathcal{B}d^-| + width(\mathcal{L}, \preceq)|\mathcal{B}d^+|$ queries.

For itemset mining, we have already the language $\mathcal{L}$ represented as sets if we consider the empty set $\{\} \in \mathcal{L}$. Therefore the *Dualize and Advance* algorithm can be applied to the problem of mining frequent itemsets. However, the language for mining sequential patterns could not satisfy the requirement of representing as sets.

3. Constraints on Representing as Sets

In this section we analyze the constraints on representing as sets. The problem of representing as sets is to find a invertible mapping function f, with which we have the structure imposed on the language $\mathcal{L}$ by $\preceq$ being isomorphic to a powerset $\mathcal{P}(\mathcal{R})$, where $\mathcal{R}$ is an finite set. This problem restricts the application of transversal hypergraph enumeration in data mining.

If an invertible mapping function $f : S \to \mathcal{P}(\mathcal{R})$ exists, the size of S must be a power of 2. For the problem of finding frequent itemsets, given relation R of items, let S be the set of all subsets of R defined by the language $\mathcal{L}$. If we consider $\{\}$ as a subset of R, then $S = \mathcal{P}(\mathcal{R})$ is a powerset and $|S| = 2^{|R|}$, thus an identity mapping $f(S) = S$ can be used in this case.

Given a data set on relation R of items, let $\mathcal{L}_{set}$ be the language of describing all itemsets generated from R and $\mathcal{L}_{seq}$ be a class of sentences that defines all sequences generated from R where the number of sentences depends on the maximal length of sentence. Obviously the number of sequences defined by the language $\mathcal{L}_{seq}$ is not a power of 2, thus $\mathcal{L}_{seq}$ cannot be mapped to a powerset by any invertible function.

Furthermore, if there exists an invertible function maps $\mathcal{L}_{seq}$ to $\mathcal{L}_{set}$, for example functions g and g^{-1}, then let $h = g \circ f$ we have that the language $\mathcal{L}_{seq}$ can be mapped to a powerset by h. In fact it does not exist such an invertible function h. So that we have the following two properties.

Property 1. *There does not exist any invertible mapping function that maps the language $\mathcal{L}_{seq}$ to a powerset.*

Property 2. *It does not exist any invertible function that maps the language $\mathcal{L}_{set}$ to language $\mathcal{L}_{seq}$ of defining sequences.*

The language $\mathcal{L}_{seq}$ of defining sequences does not satisfy the conditions on representing as sets, it does not exist any mapping function maps $\mathcal{L}_{seq}$ to a powerset.

The goal of representing as sets is to apply the transversal hypergraph enumeration to a powerset $\mathcal{P}(\mathcal{R})$ and then to retransfer the results from the powerset $\mathcal{P}(\mathcal{R})$ back to the description language $\mathcal{L}$ via the inverse function f^{-1}. This invertible transformation is based on the isomorphism on the specialization relation $\preceq$ over the mapping function f.

Property 3. *Mapping function f must be bijective.*

If a mapping function f has not the inverse function f^{-1}, after computing the transversal hypergraph, a set $X_\varphi \in \mathcal{P}(\mathcal{R})$ may not have an inverse mapping to be applied in the transformation from $X_\varphi \in \mathcal{P}(\mathcal{R})$ to the language $\varphi \in \mathcal{L}$.

In particular, for itemsets we have $\mathcal{L} \equiv \mathcal{P}(R)$, thus for all $f(\varphi) = X_\varphi$ we have $\varphi = X_\varphi$ and therefore we can simply write the identity mapping as $f(X) = X$.

Property 4. *Mapping function f must be isomorphic to the specialization relation $\preceq$.*

The mapping function for representing as sets transfers the language $\mathcal{L}$ to a powerset $\mathcal{P}(\mathcal{R})$ and then the transversal hypergraph enumeration on $\mathcal{P}(\mathcal{R})$ can be used to reduce the complexity of computing the negative border of the theory $\mathcal{T}h(\mathcal{L}, \mathbf{r}, q)$. The isomorphism requires f is monotone with respect to the specialization relation $\preceq$, that is,

$$\varphi \preceq \theta \iff f(\varphi) \preceq f(\theta).$$

4. Representing Sequences as Sets

Given a data set $\mathbf{r}$ over n rows of relation R of items, we say that a *pattern* is an itemset $X \subseteq R$. Let $\mathcal{L}_R$ denote the language defining all subsets of R, a pattern can be uniquely defined by a sentence in the language $\mathcal{L}_R$. Without losing generality, we denote the pattern as I_φ corresponding to the sentence $\varphi \in \mathcal{L}_R$. The language $\mathcal{L}_R$ describes the powerset of R if we consider the empty set $\{\}$ as a part of $\mathcal{L}_R$. The size of $\mathcal{L}_R$ is therefore $|\mathcal{L}_R| = 2^{|R|}$.

We define the *ordered pattern* as a pair (I_φ, o) where $I_\varphi \subseteq R$ is a pattern and $1 \leq o \leq n$ is an integer, the row number of the pattern. We call o the *order* of an ordered pattern. Therefore an ordered pattern is a pattern associated with an order, it can be rewritten as follows,

$$(I_\varphi, o) = \{(R_1, o), (R_2, o), \ldots, (R_j, o)\},$$

where $R_1, R_2, \ldots, R_j \in R$ and $|I_\varphi| = j$. The pair (R_i, o) is an *ordered item*, where $R_i \in R$. Let R_A denote the set of ordered items on R, we have $R_A = \{(R_i, o) \mid R_i \in R, 1 \leq o \leq n\}$, and $|R_A| = |R| \cdot n$.

Let $\mathcal{L}_A$ denote the language defining all subsets of R_O, it can be defined as following,

$$\mathcal{L}_A = \mathcal{P}(\{(X, o) \mid X \in R, 1 \leq o \leq n\}).$$

The size of $\mathcal{L}_A$ is $|\mathcal{L}_A| = 2^{n \cdot |R|}$.

We have the following characteristics of ordered patterns.

- **Union**: $I_\gamma = I_\varphi \cup I_\theta \iff (I_\gamma, o) = (I_\varphi, o) \cup (I_\theta, o)$.
- **Inclusion**: $I_\varphi \subseteq I_\theta \iff (I_\varphi, o) \subseteq (I_\theta, o)$.
- **Incomparability**: $i \neq j \Rightarrow (I_\varphi, i) \neq (I_\varphi, j)$.
- **Equivalence**: $\{(R_i, i), (R_j, j)\} = \{(R_j, j), (R_i, i)\}$.

When we consider the pair (R_i, o) as a single item, to find all frequent ordered patterns is the same task as finding frequent itemsets.

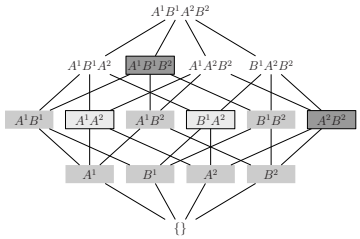

Figure 2. Finding frequent ordered patterns.

Example 1. Given data set r with items $R = \{A, B\}$, let language $\mathcal{L}^2$ define all ordered patterns with the order $o \leq 2$, we depict the language $\mathcal{L}^2$ as a lattice shown in Figure 2. Assume a set of sentences $\mathcal{S} \subseteq \mathcal{L}^2_a$ closed downwards to the relation $\preceq$, $\mathcal{S} = \{A^1, B^1, A^2, B^2, A^1B^1, A^1B^2, B^1B^2, A^2B^2, A^1B^1B^2\}$, and $\mathcal{S}$ includes the maximal ordered pattern sets $\{A^1B^1B^2, A^2B^2\}$.

The negative border $\mathcal{B}d^-(\mathcal{S}) = \{A^1A^2, B^1A^2\}$. For this problem, we already have $\mathcal{L}^2_a$ represented as sets and the mapping function f is an identity mapping. With the application of hypergraph transversals, we have therefore $\mathcal{B}d^+(\mathcal{S}) = \{A^1B^1B^2, A^2B^2\} \Rightarrow \mathcal{H}(\mathcal{S}) = \{A^2, A^1B^1\}$, thus we have the minimal transversals of $\mathcal{H}(\mathcal{S})$ that $\mathcal{T}r(\{A^2, A^1B^1\}) = \{A^1A^2, B^1A^2\}$, and thus the application of hypergraph transversals returns the correct answer. □

We use the *sequential relation* between patterns in a sequence. The sequential relation is a total order $\mapsto^o$ that a pattern I_φ is precedent to another pattern I_θ if $I_\varphi \mapsto^o I_\theta$. Let o denote the order of the sequential relation, defined as follows: given a sequence s with length of k, if for no I_γ in s we have $I_\gamma \mapsto^o I_\varphi$, then $o = 1$; otherwise, $o = max(\{o' \mid I_\theta \mapsto^{o'} I_\varphi\}) + 1$. Note that $I_\theta \neq I_\varphi$ is not required for computing $I_\theta \mapsto^{o'} I_\varphi$. In particular, for a sequence with length k, we define $o = k$ if for no I_γ in s we have $I_\varphi \mapsto^o I_\gamma$, and the order o is therefore an integer such that $1 \leq o \leq k$.

Given data set over relation R of items, let s^k denote a sequence consists of k patterns, then s^k can be formally described as follows.

$$s^k = \langle I_{\varphi_1} \mapsto^1 I_{\varphi_2} \mapsto^2 \ldots \mapsto^{k-1} I_{\varphi_k} \mapsto^k \emptyset \rangle,$$

where $I_{\varphi_1}, I_{\varphi_2}, \ldots, I_{\varphi_k} \subseteq R$ are k patterns. We use an empty set to bound a sequence. If we consider a pattern and

its followed sequential relation as a pair, such as $(I_{\varphi_i}, \mapsto^o)$, then we can represent a sequence as a set of pairs, that is,

$$s^k = \{(I_{\varphi_1}, \mapsto^1), (I_{\varphi_2}, \mapsto^2), \ldots, (I_{\varphi_k}, \mapsto^k)\},$$

where the trailing empty set can be safely removed. It is easy to see that the above form of sequence can be represented by k ordered patterns, such as,

$$s^k = \{(I_{\varphi_1}, 1), (I_{\varphi_2}, 2), \ldots, (I_{\varphi_k}, k)\}.$$

Definition 2 (Sequence)**.** *A sequence can be represented by a set of ordered patterns with consecutive orders starting from 1.*

Now let us consider a language $\mathcal{L}_O$ of generally defining all ordered patterns over relation R of items and given maximal order n, without distinguishing the form of representation. Let $\mathcal{L}_{seq}$ denote the language of defining all sequences over attributes R with given maximal length n, we have $\mathcal{L}_{seq} \subset \mathcal{L}_O$. Semantically, under the context of transaction database, we have following properties.

Property 5 (From Ordered Patterns to Sequence)**.** *Each non-empty sentence in $\mathcal{L}_O$ stands for a list of transactions with their order in transaction time, corresponding to a sequence. Multiple sets of ordered patterns can be represented as one sequence in semantics.*

Property 6 (From Sequence to Ordered Patterns)**.** *Each non-empty sentence in $\mathcal{L}_{seq}$ stands for a sequence of transaction, which can be only represented by a set of ordered patterns with consecutive orders starting from 1.*

Therefore, a *production function* p can be used in transforming a set of ordered patterns to a sequence. Given a sentence $\varphi \in \mathcal{L}_O$, such as,

$$\varphi = \{(I_{\varphi_1}, o_1), (I_{\varphi_2}, o_2), \ldots, (I_{\varphi_k}, o_k)\},$$

where $o_1 < o_2 < \ldots < o_k$ and $k \leq n$. The production $p(\varphi)$ returns a new sentence $\theta \in \mathcal{L}_O$ such that,

$$\theta = \{(I_{\varphi_1}, 1), (I_{\varphi_2}, 2), \ldots, (I_{\varphi_k}, k)\}.$$

We say that the sentence θ is the *alias* of the sentence φ, which represents a sequence.

It is remarkable that the production function p is not invertible, so that it does not imply that this representation satisfies the requirement of representing as sets.

5. Mining Sequential Patterns with Transversal Hypergraph Enumeration

We propose the *HSP* algorithm for mining sequential patterns with transversal hypergraph enumeration. Given a transactional database over relation R, the task of mining sequential patterns is to find maximal frequent sequences with respect to given *minimal support* [3]. We use the language $\mathcal{L}_A$ for representing sequences and use the *Dualize and Advance* algorithm in finding the positive border of all interesting sentences of $\mathcal{L}_A$. The sequential pattern mining process is specified within the *Dualize and Advance* algorithm by a predicate q_hsp that determines whether the sequence corresponded to each sentence is frequent or not. This procedure returns all most specific sentences, i.e., all most specific frequent ordered patterns and their aliases. Finally the *HSP* algorithm returns all frequent sequential patterns with respect to these aliases.

Due to the limit of space, this paper only introduce the algorithm of q_hsp that is defined as follows (shown as Algorithm 1). Given a set $\mathcal{S}$ of customer sequences and a sentence $\varphi \in \mathcal{L}_A$, q_hsp evaluates φ against each $s \in \mathcal{S}$. If φ does not exist in any s, q_hsp returns $false$ without further evaluations. Otherwise, q_hsp computes the alias $\theta \in \mathcal{L}_A$ of φ and expands θ to obtain all sentences $\mathcal{E}$ ($\varphi \notin \mathcal{E}$) having the same alias θ. q_hsp then evaluates $\tau \in \mathcal{E}$ in each customer sequence, and updates the rank of θ for computing the support of θ. If the support of θ is $\geq minimal\_support$ q_hsp stores θ as a frequent sequence and returns $true$ otherwise q_hsp returns $false$. The approach of *Dualize and Advance* requires that q_hsp is monotone to the specialization relation $\preceq$ on the language $\mathcal{L}_A$.

Property 7. *The predicate q_hsp is monotone to the specialization relation $\preceq$ on the language $\mathcal{L}_A$.*

Proof. We already have that the sentences of $\mathcal{L}_A$ respect the specialization $\preceq$. If a sentence $\varphi \in \mathcal{L}_A$ is interesting, then we have the alias $\theta \in \mathcal{L}_A$ interesting, means that the sequence s_θ represented by θ is frequent, and φ exists in at least one customer sequence $s \in \mathcal{S}$. And according to the relation $\preceq$ on $\mathcal{L}_A$, any generalization of φ must be exist in at least one customer sequence $s \in \mathcal{S}$, and any sub-sequences of s_θ must be frequent. Thus we have that for $\gamma \in \mathcal{L}_A$ and $\gamma \preceq \varphi$, if $q\_hsp(\mathcal{S}, \varphi) = true$, then $q\_hsp(\mathcal{S}, \gamma) = true$.

Next we show that for $\gamma \in \mathcal{L}_A$ and $\gamma \preceq \varphi$, if $q\_hsp(\mathcal{S}, \gamma) = false$, then $q\_hsp(\mathcal{S}, \varphi) = false$.

$q\_hsp(\mathcal{S}, \gamma) = false$ means that γ does not exist in any customer sequence $s \in \mathcal{S}$ or the sentence represented by the alias $\psi \in \mathcal{L}_A$ of γ is not frequent. In the first case, no specialization of γ can exist in any customer sequence $s \in \mathcal{S}$. In the second case, the sequence represented by the alias of any specialization of γ cannot be frequent. Thus for any sentence $\varphi \in \mathcal{L}_A$ and $\gamma \preceq \varphi$, we have $q\_hsp(\mathcal{S}, \varphi) = false$. □

The predicate q_hsp is monotone to the specialization relation $\preceq$ on $\mathcal{L}_A$ and it updates correctly the frequency of sequences with respect to the definition of *support* for sequential patterns. Therefore the model of ordered patterns

and the q_hsp predicate can address the problem of mining sequential patterns within the *Dualize and Advance* algorithm where the transversal hypergraph enumeration is applicable.

Algorithm 1: Algorithm of the *q_hsp* predicate.

if *exits* $\gamma \preceq \varphi$ *not interesting* **then**
  return false;
end
$alias\_rank \leftarrow 0$;
foreach $s \in \mathcal{S}$ **do**
  $rank \leftarrow$ evaluate φ against s;
  if $rank > 0$ **then**
    update $alias\_rank$ by $rank$;
    remove s from $\mathcal{S}$;
  end
end
if $alias\_rank = 0$ **then**
  return false;
end
$\theta \leftarrow$ alias of φ;
$\mathcal{E} \leftarrow$ all sentences with alias θ but excluding φ;
foreach $s \in \mathcal{S}$ **do**
  foreach $\tau \in \mathcal{E}$ **do**
    $rank \leftarrow$ evaluate τ against s;
    if $rank > 0$ **then**
      update $alias\_rank$ by $rank$;
      remove s from $\mathcal{S}$;
    end
  end
end
if $alias\_rank/number\_of\_slices \geq min\_supp$ **then**
  store_alias(θ);
  return true;
end
return false;

According to the *Dualize and Advance* algorithm, the complexity of the *HSP* is polynomial in $|\mathcal{B}d^+|$ and $T(|\mathcal{B}d^+|, |\mathcal{B}d^-|)$ where $T(n) = n^{\mathcal{O}(\log(n))}$ [5, 6, 7].

Given a set $\mathcal{S}$ of customer sequences, assume the final result contains N aliases, then in the worst case the number of all most specific sentences is $|\mathcal{B}d^+| = N|\mathcal{S}|$. In the worst case, each evaluation of a sentence $\varphi \in \mathcal{L}_A$ requires $|\mathcal{S}| + |\varphi|(|\mathcal{S}| - 1)$ queries without caching, where $|\varphi|$ is the number of all sentences with the same alias with φ, not including φ, and $|\mathcal{S}|$ is the number of customer sequences.

6. Conclusion

We analyzed the constraints of using transversal hypergraph enumeration on itemset mining and proposed the *HSP* approach for mining sequential patterns with the *Dualize and Advance* algorithm. We introduced the model of ordered patterns with respect to the constraints on applying transversal hypergraph enumeration in itemset mining. We showed that the problem of mining sequential patterns could be addressed by the language $\mathcal{L}_A$ for finding frequent ordered patterns and we presented a predicate for determining whether the sequence corresponded to each sentence is frequent. This predicate is monotone on $(\mathcal{L}_A, \preceq)$. The *HSP* approach is interesting when the lengths of frequent sequential patterns are large. We are currently investigating the comparison between different approaches to sequential patterns mining.

References

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *SIGMOD Conference*, pages 207–216, 1993.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*, pages 487–499, 1994.

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In *ICDE*, pages 3–14, 1995.

[4] C. Berge. *Hypergraphs*. Combinatorics of Finite Sets. North-Holland, Amsterdam, 3rd edition, 1973.

[5] M. L. Fredman and L. Khachiyan. On the complexity of dualization of monotone disjunctive normal forms. *J. Algorithms*, 21(3):618–628, 1996.

[6] D. Gunopulos, R. Khardon, H. Mannila, S. Saluja, H. Toivonen, and R. S. Sharm. Discovering all most specific sentences. *ACM Trans. Database Syst.*, 28(2):140–174, 2003.

[7] D. J. Kavvadias and E. C. Stavropoulos. Evaluation of an algorithm for the transversal hypergraph problem. In *Algorithm Engineering*, pages 72–84, 1999.

[8] H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Min. Knowl. Discov.*, 1(3):241–258, 1997.

[9] H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. In *KDD Workshop*, pages 181–192, 1994.