Research on Typical Algorithms in Negative Sequential Pattern Mining

Yongshun Gong, Chuanlu Liu and Xiangjun Dong*

School of Information, Qilu University of Technology, Shandong, Jinan, 250353, P.R. China

Abstract: Negative sequential pattern (NSP), which contains both non-occurring and occurring items, is sometimes much more informative than positive sequential pattern (PSP) in many applications and intelligent systems. To date, very limited methods are available to mine NSP due to its intrinsic complexities. Furthermore, there is not a unified definition about negative containment, i.e., how a data sequence contains a negative sequence. The researchers who begin to study negative sequential pattern are often confused by these different definitions and are eager to know the differences among the existing methods. So in this paper, we select four typical existing methods, PNSP, Neg-GSP, e-NSP and NSPM, implement their algorithms by JAVA, and compare their definitions, methods, runtimes and the number of NSPs. Examples and experiments on four datasets clearly show their differences.

Keywords: Negative containment, Negative sequential pattern, Positive sequential pattern.

1. INTRODUCTION

Sequential pattern mining has developed rapidly since proposed in 1995 [1]. Some well-known algorithms, such as GSP [2], FreeSpan [3], PrefixSpan [4], SPADE [5] and SPAM [6], have been proposed to improve the efficiency of mining sequential patterns. These patterns, focusing only on occurring items, are called positive sequential patterns (PSPs) in contrast to negative sequential patterns (NSPs) that consist of both occurring and non-occurring items [7, 8]. NSP, which includes at least one non-occurring item, such as <*a*(*ab*)¬*cac*>, can play more important role than PSP in some applications. For instance, assume p_1=<*abcX*> is a PSP; p_2=<*ab*¬*cY*> is a NSP, where *a*, *b* and *c* stand for medical service codes that a patient has received in health care, and *X* and *Y* stand for disease status. p_1 shows that a patient who usually receives medical services *a*, *b* and then *c* is likely to have disease status *X*, whereas p_2 indicates that patients receiving treatment of *a* and *b* but NOT *c* have a high probability of having disease status *Y*. Such NSP becomes increasingly important, and can play an irreplaceable role in tackling many business applications, such as the associations between treatment services and illnesses [7, 8]. For another example, in social welfare, the lack of follow-up examination after the address change of a customer may lead to overpayment to the customer; for a credit card company, it is interesting to find the positive and negative relationships between transaction sequences and an unrecovered debt [9].

Very few methods are available to NSP mining so far, because mining NSP is much more challenging than mining PSP [8]. Furthermore, there is not a unified definition about negative containment, i.e., how to define that weather a data sequence contains a negative sequence. For example, whether data sequence <*eb*> contains negative sequences <*e*¬*e*>, <¬*ee*>, and <¬*ee*¬*e*>, whether <*ac*> contains <*a*¬*bc*¬*d*>, and whether <*aycxe*> or <*axcde*> contains <*a*¬*bc*¬*de*>. To date, different definitions and explanations have been proposed in some relevant papers, such as PNSP [7], Neg-GSP [8], e-NSP [10], NSPM [11] and [12], we will discuss details in the rest sections.

The researchers who just touched on negative sequential patterns are often confused by these different definitions of negative containment. They are eager to know the differences of existing methods. So in this paper, we choose four typical methods PNSP [7], Neg-GSP [8], e-NSP [10] and NSPM [11] to compare their definitions and mining methods by four examples on the same sample database. And we also conducted intensive experiments on the four datasets to show their efficiency and mining ability. Examples and experiments clearly show their differences.

The rest of this paper is organized as follows. The related work is described in detail in Section 2. Section 3 discusses the comparisons. Section 4 shows the experiment results, which are followed by conclusions and future work in Section 5.

2. RELATED WORK

Zheng *et al.* proposed an algorithm, Neg-GSP, to mining NSPs [8]. In order to adapt to negative conditions, Neg-GSP improves the approaches of generating and pruning candidates in GSP. The general idea of Neg-GSP is to mine PSPs by GSP first, then to generate and prune NSCs, and then to count the support of NSCs by scanning database to get all defined negative patterns. Some measures are also proposed to prune unnecessary NSCs in Neg-GSP. [13] proposed a genetic algorithm based approach to mine NSPs. The approach borrows idea of gene evolution, to get candidates by crossover and mutation. It proposed dynamic fitness function to generate as many candidates as possible and to avoid population stagnating.

*Address correspondence to this author at the School of Information, Qilu University of Technology, Shandong, Jinan, 250353, P.R China;
Tel: +86-531-89631336; E-mail: d-xj@163.com.

PNSP is a NSP mining method proposed by Chen *et al.* for mining positive and negative sequential patterns in the form of $<(abd)\neg(de)(ij)>$ [7]. Their approach is broken into three phases. First, it finds all positive sequential patterns. Second, all positive itemsets are found, from which all negative itemsets are derived. Third, both positive and negative itemsets are in turn joined iteratively to generate longer negative sequential candidates based on an Apriori-like way. Then, the supports of these candidates are calculated by re-scanning the database.

Ouyang *et al.* proposed a method to mine negative sequential patterns as $(\neg A,B)$, $(A, \neg B)$ and $(\neg A,\neg B)$ [12], which is similar to mine negative association rules. The method requires $A\cap B=\varnothing$, which is a usual constraint to association rule mining but is a very strict constraint to sequential pattern mining. The core idea of this method is to find frequent itemsets first of all, then use these to generate frequent and infrequent sequences and finally negative sequential patterns are derived from infrequent sequences. [14-16] are three extended versions of [12] by adding fuzzy, multiple level, multiple minimum support constraints respectively. [17] mentioned the question of mining NSP, but hasn't given a specific method of how to mine it. [18] mined NSP in the same form as [12] in incremental transaction databases.

Lin *et al.* proposed an algorithm NSPM (Negative Sequential Patterns Mining) for mining negative sequential patterns, which only handles NSP as the last element can be negative, and all other elements are positive [11]. They extended their algorithm to NFSPM for mining negative fuzzy sequential patterns [19], and PNSPM for mining strong positive and negative sequential patterns [20].

e-NSP is an efficient algorithm of mining NSP without re-scanning database [10]. Firstly, all PSPs are mined from the sequence database by using PSP mining algorithm. Secondly, NSCs are generated by changing any non-contiguous elements (not items) in a PSP to their negative ones. Then the supports of NSC can be calculated efficiently by only using the NSC's corresponding PSP information without re-scanning the database.

3. COMPARISONS

3.1. Problem Statements

Assume $I = \{i_1, i_2\dots i_n\}$ is a set of items. An itemset is a subset of items. A sequence is an ordered list of itemsets. A sequence s is denoted as $<s_1s_2\dots s_l>$, where $s_j \subseteq I$ for $1\leq j\leq l$. s_j is also called an element of the sequence, and denoted as $(x_1 x_2\dots x_m)$, where x_k is an item, i.e., $x_k\in$ I for $1\leq k\leq m$. For brevity, the brackets are omitted if an element has only one item. That is, *element* (x) is written as x. An item can occur at most once in an element of a sequence, but can occur at multiple times in different elements of a sequence. *length*(s) is the length of sequence$s$, which is the total number of items in all the elements in s. Sequence s is a *k-length* sequence if *length*(s)=k. The size of sequence s, denoted as *size*(s), is the total number of elements in s. Sequence s is a *k-size* sequence if *size*(s)= k. For example, s_1=$<acef>$ is a *4-length*, *4-size* sequence; s_2=$<(af)cd>$ and s_3=$<a(bc)$e$>$ are both *4-length*, *3-size* sequences.

Sequence α= $<\alpha_1\alpha_2 \dots \alpha_n>$ is called a subsequence of sequence $\beta = <\beta_1 \beta_2 \dots \beta_m>$and β a super sequence of α, denoted as $\alpha\sqsubseteq\beta$, if there exist integers $1 \leq j_1 < j_2 < \dots < j_n\leq m$ such that $\alpha_1 \subseteq \beta_{j_1}, \alpha_2 \subseteq \beta_{j_2},\dots,\alpha_n \subseteq \beta_{j_n}$. We also say β contains α. For example, $<b(ce)>$ is a subsequence of $<bb(cde)>$, $<bb(cef)>$ is super subsequence of $<b(cf)>$, or $<bb(cef)>$ contains $<b(cf)>$.

A sequence database D is a set of tuples $<sid, ds>$, where *sid* is a sequence_id and *ds* is a data sequence. The number of tuples in D is denoted as $|D|$. The set of tuples containing sequence α is denoted as $\{<\alpha>\}$. The support of α, denoted as $sup(\alpha)$, is the number of $\{<\alpha>\}$, i.e., $sup(\alpha)$= $|\{<\alpha>\}|$=$|\{<sid, ds>| <sid, ds> \in D \wedge (\alpha \sqsubseteq ds)\}|$. The *min_sup* is a minimum support threshold given by users or experts. Sequence α is called a (positive) sequential pattern, or α is frequent, if $sup(\alpha)\geq min\_sup$. The task of (positive) sequential pattern mining is to discover the set of all (positive) sequential patterns.

A sequence s=$<s_1s_2\dots s_l>$, is a positive sequence, when each element s_j, $1\leq j\leq l$ is a positive element. A sequence s=$<s_1s_2\dots s_l>$, is a negative sequence, when $\exists j$, s_j, $1\leq j\leq l$, s_j is a negative element, which represents a non-occurring element. Sequence s is called a negative sequential pattern, or s is frequent, if $sup(s)\geq min\_sup$. The task of negative sequential pattern mining is to discover some sub-set (if not all) of negative sequential patterns.

In the following section, we will introduce definitions of negative containment in PNSP, Neg-GSP and e_NSP and discuss their mining strategies based on the same sample database which is shown in Table **1**.

3.2. Definition and Method in PNSP

To understand the negative containment definition of PNSP, we should first introduce two definitions: *p-contain* and *n-cover*, which is important for the explanation.

Positive sequence β *p-contains* positive sequence α only if α is a subsequence of β. A data sequence ds=$<d_1d_2\dots d_m>$ *n-covers* a negative sequence ns =$<e_1e_2\dots e_k>$ if there exist integers p, q, r and $e_i \in \neg I$ in ns such that two conditions hold:

(1) *ds p-contains ns.mp* (the maximum positive subsequence of *ns* is denoted as *ns.mp*).

(2) $\exists e_{i-1}\subseteq d_q \wedge e_{i+1}\subseteq d_r \wedge e_i\not\subset\forall d_q$. $(1\leq i\leq k, 1\leq p<q<r)$

The negative containment in PNSP is as follows:

Data sequence *ds n-contains* negative sequence *ns* supposing that *ds n-covers ns* while *ds* does not *p-contain ns.ce* (the counter example of a negative *ns* is denoted as *ns.ce*. For example, the *counter example* of $<c\neg(ad)b\neg d>$ is $<c(ad)bd>$.

Table 1. Sample database.

Sid	Sequence
D1	<*abc*>
D2	<*a*(*ab*)>
D3	<(*ae*)(*ab*)*c*>
D4	<*aa*>
D5	<*d*>

Table 2. Positive sequential patterns mined by GSP.

1-length Sequences	*2-length* Sequences	*3-length* Sequences
<*a*>	<*aa*>	<*abc*>
<*b*>	<*ab*>	<*a*(*ab*)>
<*c*>	<*ac*>	
	<*bc*>	
	<(*ab*)>	

Table 3. Negative sequential patterns mined by PNSP.

1-size Sequences	2-size Sequences	3-size Sequences
<¬*a*>	<¬*ba*>	<¬*bbc*>
<¬*b*>	<¬*bb*>	<¬*bb*¬*a*>
	<¬*b(ab)*>	<¬*bb*¬*b*>
	<*b*¬*a*>	<*ab*¬*a*>
	<*b*¬*b*>	<*ab*¬*b*>

Now, we introduce the mining process of PNSP algorithm. The method in PNSP is divided into three phases [7].

In the first phase, PNSP uses the traditional GSP algorithm to find all positive sequential patterns. Table **2** shows the result from database in Table **1**. Then, all positive frequent itemsets are recognized as *1-size* frequent sequences. These itemsets are used to find all the negative itemsets satisfying *miss_freq*(missing frequency threshold) in Phase 2, to generate candidate negative sequences, and to prune some candidate negative sequences.

In second phase, PNSP uses above frequent itemsets for generating negative candidate itemsets. The supports of these itemsets are greater than *miss_freq* or less than *min_sup.*

The third phase is the core of NSP mining process. *k-size* negative sequential candidates are generated by appending a positive or negative frequent itemset to a (*k-1*)-*size* candidate sequential patterns. Then, with re-scanning the database, the supports of these candidates can be calculated easily. Table **3** shows negative sequential patterns mined by PNSP from the data in Table **1** (*min_sup*:25%, *miss_freq*:40%).

3.3. Definition and Method in Neg-GSP

The main idea of Neg-GSP is similar with GSP algorithm. In Neg-GSP, it also improves joining and pruning methods for mining process. The negative containment in Neg-GSP is determined by *base-match* and *match* methods, the former is used to find seed sequences.

For all negative elements e_i in negative sequence *ns*, *ns base-matches* a data sequence *ds* if *ns is* satisfied (*p*, *q*, and *r* are three integers):

(1) The max positive subsequence of *ns* is contained by *ds*.

(2) $\exists e_{i-1} \subseteq d_q \wedge e_{i+1} \subseteq d_r$, and for $\exists d_q$, $e_i \not\subset d_q$.

Table 4. Negative sequential patterns mined by Neg-GSP.

1-length Sequences	*2-length* Sequences	*3-length* Sequences
<¬*b*>	<*b*¬*a*>	<*ab*¬*a*>
<¬*c*>	< ¬*ba*>	<*ab*¬*b*>
	<¬*bb*>	<¬*cbc*>
	<*b*¬*b*>	<*a*¬*cc*>
	<¬*ca*>	<¬*bb*¬*a*>
	<¬*cb*>	<¬*bb*¬*b*>
	<¬*cc*>	<¬*cb*¬*a*>
	<*a*¬*c*>	<¬*cb*¬*b*>

The negative sequence *ns matches* a data sequence *ds* if *ns is* satisfied:

(1) The max positive subsequence of *ns* is contained by *ds*.

(2) $\exists e_{i-1} \subseteq d_q \wedge e_{i+1} \subseteq d_r$, and for $\forall d_q$, $e_i \not\subset d_q$.

The Neg-GSP method is as follows [8].

Step 1: Neg-GSP uses GSP algorithm to mine all the positive sequential patterns (shown in Table **2**).

Step 2: All *1-length* positive patterns are transformed to *1-length* negative patterns, which will be recognized as *1-length* candidate sets. Then *1-length* negative patterns are also regarded as *1-length* seed set and *1-length* patterns.

Step 3: Use all (*k-1*)*-length* seed sequences and (*k-1*)*-length* positive patterns to generate *k-length* candidates by employing the join method. This process can be represented as:

S_{k-1} & $S_{k-1} \rightarrow C_k$ and S_{k-1} & $P_{k-1} \rightarrow C_k$

Where S_{k-1} means the *(k-1)-length* seed set, P_{k-1} means the *(k-1)-length* frequent positive sequence set, C_{k-1} means the *k-length* candidate seed set and the symbol & indicates the join method.

Step 4: In order to get a smaller candidate set, this set will be pruned by the following method: If a candidate's *base-sup* value is less than *min_sup*, the candidate should be pruned.

Step 5: Calculate the support values and *base-support* values of all candidates. If one's *base-support* value is greater than *min-sup*, this candidate will be added into *k-length* seed set. And if one candidate support value is greater than *min-sup*, then it is frequent and will be output as a *k-length* pattern.

Step 6: Check whether the *k-length* seed set is empty. If not, increase *k* by one and loop back to Step 3 until the candidates set is empty.

Table **4** shows the negative sequential patterns mined by Neg-GSP algorithm from the data in Table **1** (*min_sup*:25%).

3.4. Definition and Method in e-NSP

To understand the negative containment definition and method in e-NSP, we discuss two definitions, *p(ns)*, *MPS(ns)* at first.

For a negative sequence $ns = <s_1s_2...s_m>$, *p(ns)* can change all negative elements of *ns* to their corresponding positive elements. We can use *MPS(ns)* to find the maximum positive subsequence of *ns*. For example, *p*(<¬*a* (*cf*) ¬*d*>)=< *a* (*cf*) *d* >; *MPS*(<*a* ¬(*cf*) *d* >)=<*a d*>.

The negative containment definition in e-NSP is very complicated so that we use a negative conversion method to explain it.

Assume *ns* is a negative sequence, connect its one negative element and *MPS(ns)* as a sub-sequence of *ns*, which is called a 1-neg-size maximum sub-sequence, denoted as 1-*negMS*. For example, *ns*=<¬*a*(*bd*)*e*¬(*ij*)>, so its 1-*negMS* are <*a*(*bd*) *e*> and <(*bd*)*e*(*ij*)>.

Let $ds = <d_1d_2 ...d_t>$ be a data sequence, $ns = <s_1s_2 ... s_m>$ be an *m-size* and *n-neg-size* negative sequence. A negative sequence *ns* is contained by the data sequence *ds* if the three conditions hold [10].

(1) $m+1 \geq 2n$.

(2) $MPS(ns) \subseteq ds$.

(3) Each *1-negMS* satisfies the $p(1\text{-}negMS) \not\subset ds$.

The mining method in e-NSP algorithm is shown as follows:

Step 1: All the positive sequential patterns are mined by any PSP Mining algorithm, such as GSP, PrefixSpan, SPADE, etc. All PSPs and their *sid* sets are saved in a hash table, in which PSP's hash code acts as key code. For example, *sid* set of a positive sequence <*ab*>, is {D1, D2, D3} means <*ab*> is contained in the sequence D1, D2 and D3.

Step 2: For generating NSCs, the algorithm changed any *m* non-contiguous elements (not items) in a PSP to their negative ones. For a *k* size PSP, $m=1,2,\ldots,\lceil k/2 \rceil$. For example, a PSP *p*=<*a*(*bc*)*f*> can generate the following NSC:

Table 5. Negative sequential patterns mined by e-NSP.

1-length Sequences	*2-length* Sequences	*3-length* Sequences
<¬*b*>	<*a*¬*c*>	<*a*¬ (a*b*)>
<¬*c*>	<¬ (*ab*)>	

Table 6. Negative sequential patterns mined By NSPM.

1-length Sequences	*2-length* Sequences	*3-length* Sequences
<¬*a*>	<*b*¬*a*>	<*ac*¬*b*>
<¬*b*>	<*c*¬*a*>	<*bc*¬*a*>
<¬*c*>	<*c*¬*b*>	

m=1, <¬*a*(*bc*)*f*>, <*a*¬(*bc*)*f*>, < *a*(*bc*)¬*f*>;

m=2, <¬*a*(*bc*)¬*f*>.

Step 3: Assume *ns* is a *m-size* and *n-neg-size* negative sequence. The supports of negative sequences are calculated by the following equation.

$$sup(ns) = sup(MPS(ns)) - |\cup_{i=1}^{n}\{p(1\text{-}negMS_i)\}| \qquad (1)$$

The *sup*(*MPS*(*ns*)) can be calculated by traditional algorithms easily. The core problem is how to calculate $|\cup_{i=1}^{n}\{p(1\text{-}negMS_i)\}|$. At first step, we have got all PSPs and their *sid* sets, so we can get the *sid*'s union set from $\{p(1\text{-}negMS_i)\}$ based on the corresponding PSPs. And the number of union set is what we want.

Table **5** shows the negative sequential patterns mined by e-NSP algorithm from the data in Table **1** (*min_sup*:25%).

3.5. Definition and Method in NSPM

The negative containment definition in NSPM is as follows:

In NSPM, a negative sequence b=<$b_1b_2\ldots b_n$> is contained in a sequence s=<$s_1s_2\ldots s_m$>, if its positive counterpart <$b_1b_2\ldots b_n$> is not contained in *s* and the subsequence, <$b_1b_2\ldots b_{n-1}$>, of *b* is contained in *s*.

The method in NSPM consists of two phases [11]: positive sequential patterns mining phase and negative sequential patterns mining phase.

In the phase of mining positive sequential patterns, the positive candidates of length *k*, CP_k, are generated from joining LP_{k-1} with LP_{k-1} by *p_gen* function. Then, supports of these candidates are counted by scanning the database *D* to select large *k*-sequences, LP_k.

In the negative sequential patterns mining phase, the negative candidate sequences of length *k*, CN_k, are generated from joining LP_{k-1} with LN_{k-1} by *n_gen* function. Next, supports of these candidates are counted to determine large k-sequences LN_k. Then, the value of the interestingness measure function *im* of these large sequences is computed for finding negative sequential patterns IN_k. Finally, IN_k is added into *N* which contains all negative patterns.

Table **6** shows the negative sequential patterns mined by NSPM algorithm from the data in Table **1** (*min_sup*:25%).

3.6. Comparisons

From Table **3** to Table **6**, we can see that negative sequential patterns mined by the three algorithms are different. The main reason is the various definitions of negative containment in these algorithms. For example, PNSP algorithm uses the concept of *miss_freq*, Neg-GSP algorithm uses the concept of *base-match* and *match*, and NSPM algorithm uses the concept of *n_gen* function, to judge whether the 1-*lengeh* negative candidate is frequent or not.

For this reason, in PNSP algorithm, the supports of <¬*c*>, <¬*d*> and <¬*e*> are greater than the *miss_freq*, so these 1-length negative candidates should be pruned. And <¬*a*>, <¬*b*> are output as 1-*length* negative frequent sequences.

In Neg-GSP algorithm, only the supports of <¬*b*> and <¬*c*> are greater than *min-sup*, then these candidates are frequent and are output as 1-*length* negative frequent sequences.

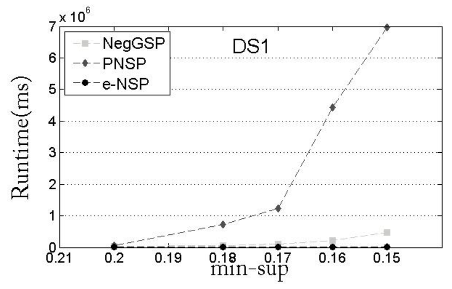
In NSPM algorithm, based on the *n_gen* function, <¬*a*>, <¬*b*> and <¬*c*> are discovered as *1-length* negative frequent sequences.

Different 1-*length* negative frequent sequences of three NSP algorithms lead to different 2-*length* negative frequent sequences and 3-*length* negative frequent sequences consequently.

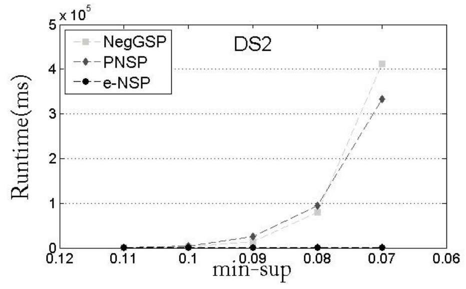
Now, let's look at the answers of the questions in Section 1.

In PNSP, <*eb*> contains negative sequences <*e*¬*e*>, but does not contain <¬*ee*>, <¬*ee*¬*e*>. <*ac*> does not contain <*a*¬*bc*¬*d*>. <*aycxe*> contains <*a*¬*bc*¬*de*>, and <*axcde*> does not contain <*a*¬*bc*¬*de*>.

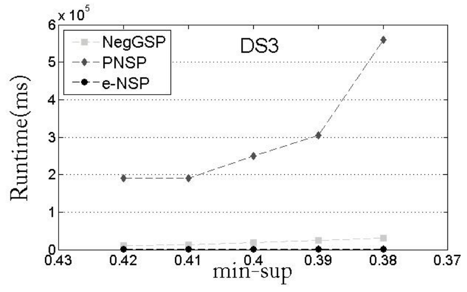
In Neg-GSP, <*eb*> contains negative sequences <*e*¬*e*>, but does not contain <¬*ee*>, <¬*ee*¬*e*>. <*ac*> does not contain <*a*¬*bc*¬*d*>. <*aycxe*> contains <*a*¬*bc*¬*de*>, and <*axcde*> does not contain <*a*¬*bc*¬*de*>.



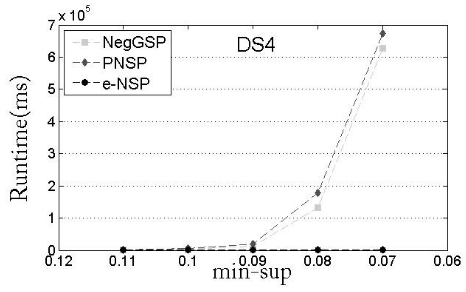
(**a**) Runtime on DS1



(**b**) Runtime on DS2



(**c**) Runtime on DS3



(**d**) Runtime on DS4

Fig. (1). Runtime on four datasets.

In e_NSP, <*eb*> contains negative sequences <*e*¬*e*>, <¬*ee*>, <¬*ee*¬*e*>. <*ac*> contains <*a*¬*bc*¬*d*>. <*aycxe*> contains <*a*¬*bc*¬*de*>, and <*axcde*> does not contain <*a*¬*bc*¬*de*>.

These questions have no answer in NSPM because NSPM only allows the last element to be negative and others are positive.

4. EXPERIMENTS

In this section, we will compare the above algorithms' runtime and the number of NSPs. Because algorithm NSPM only allows that the last element of a NSP is negative, it has no comparability with other algorithms. Thus, our comparisons are only among e-NSP, PNSP and Neg-GSP.

We conducted experiments on four datasets and all algorithms are implemented using MyEclipse 8.5 in a PC with Pentium 4 Celeron 2.1G PC and 2G main memory, Windows 7 Professional.

4.1. Datasets

We use the IBM data generator [1] to generate datasets.

Dataset1(*DS1*) is C4_T4_S6_I6_DB10k_N100;

Dataset2(*DS2*) is C6_T4_S4_I4_DB5k_N100;

Dataset3(*DS3*) is C8_T4_S6_I6_DB10k_N100;

Dataset3(*DS4*) is C6_T4_S4_I4_DB10k_N100;

For example, *DS1* means the dataset has 10000 sequences and the number of items is 1000 (because the default unit of N is 10). On average, the number of elements in a sequence is 4, and the number of items in each element is 4. The average length of maximal pattern consists of 6 elements and each element is composed of 6 items average [8].

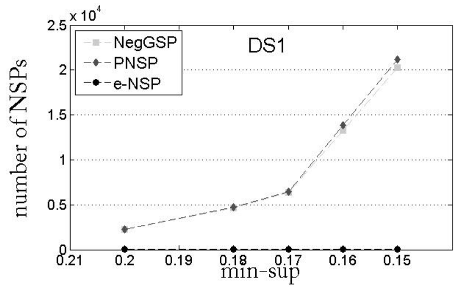
4.2. Experimental Results

Fig. (**1**) shows the runtimes of three algorithms on different datasets. Fig. (**2**) shows the number of negative patterns mined by three algorithms. From Fig. (**1**) we can see that the runtime of e-NSP is lower than the other two algorithms, but from Fig. (**2**) we can see that the number of NSPs mined by e-NSP is also less than other two algorithms. Therefore, we can't simply judge their efficiency by only comparing their runtime. Therefore, we use the runtime per pattern to describe their efficiency. It can be written as runtime/number of NSPs.

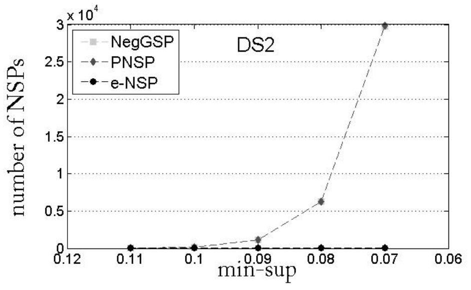
From Fig. (**3**) we can see that e-NSP is more efficient than PNSP and Neg-GSP.

5. CONCLUSION AND FUTURE WORK

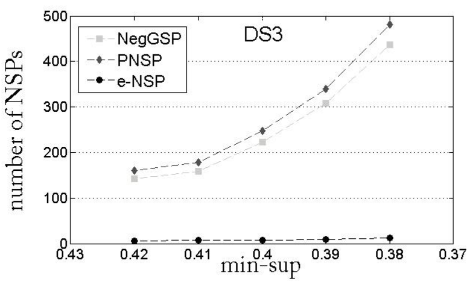
Negative sequential patterns including at least one non-occurring item, can play an important role in many intelligent systems and business applications. NSP mining is at an early stage, and has seen only very limited progress in recent years. In order to know the differences between existing methods, we have compared four typical methods, PNSP, Neg-GSP, e-NSP and NSPM through examples and conducted experiments on first three algorithms because NSPM only allows the last element to be negative. The experimental results show that the runtime of e-NSP is lower than other two algorithms, but the number of NSPs mined by e-NSP is also less than the other two algorithms. Therefore, we use the runtime per pattern to describe their efficiency. It can write as runtime/number of NSPs and through the experiments we



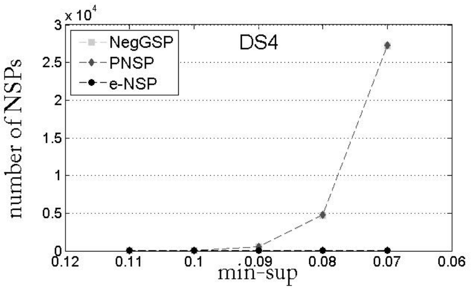
(**a**) Number of NSPs on DS1



(**b**) Number of NSPs on DS2

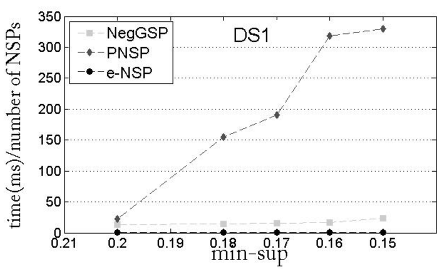


(**c**) Number of NSPs on DS3

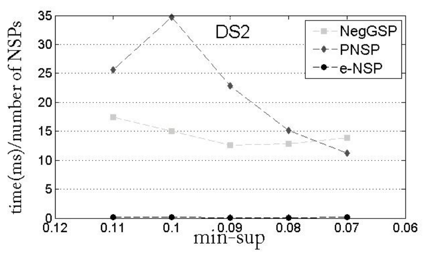


(**d**) Number of NSPs on DS4

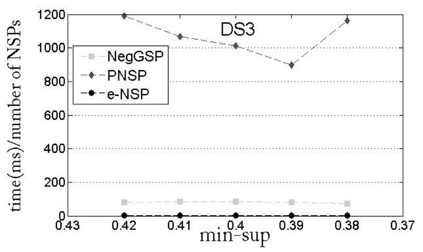
Fig. (2). Number of NSPs on four datasets.



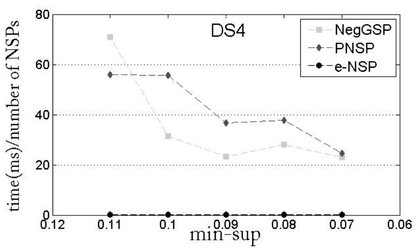
(**a**) Runtime/number of NSPs on DS1



(**b**) Runtime/number of NSPs on DS2



(**c**) Runtime/number of NSPs on DS3



(**d**) Runtime/number of NSPs on DS4

Fig. (3). Runtime/number of NSPs on four datasets.

can find that e-NSP is much more efficient than PNSP and Neg-GSP.

For future work, we will try to set up a unified definition of negative containment and propose a new method to mine negative sequential pattern efficiently.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

This work was supported partly by National Natural Science Foundation of China (71271125), Natural Science Foundation of Shandong Province, China (ZR2011FM028) and Scientific Research Development Plan Project of Shandong Provincial Education Department (J12LN10).

REFERENCES

[1] R. Agrawal and R. Srikant, "Mining sequential patterns," In: *Proceedings of the 11th International Conference on Data Engineering, IEEE Computer Society Press*, Taipei, Taiwan, 1995, pp. 3-14.

[2] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," In: *EDBT '96 Proceedings of the 5th International Conference on Extending Database Technology*, London, UK, 1996, pp. 3-17.

[3] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M. C. Hsu, "Freespan: frequent pattern-projected sequential pattern mining," In: *KDD '00: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2000, pp. 355–359.

[4] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu, "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth," In: *ICDE '01: Proceedings of the 17th International Conference on Data Engineering, IEEE Computer Society*, Washington, DC, USA, 2001, pp. 215-226.

[5] M.J. Zaki, "Spade: An efficient algorithm for mining frequent sequences," *Machine Learning*, vol. 42, pp. 31-60, 2001.

[6] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu. "Sequential pattern mining using a bitmap representation." In: *KDD'02: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2002, pp. 429-435.

[7] S.C. Hsueh, M.Y. Lin, and C.L. Chen, "Mining Negative Sequential Patterns for E-commerce Recommendations," In: *Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference,* IEEE Computer Society, 2008, pp. 1213-1218.

[8] Z. Zheng, Y. Zhao, Z. Zuo, and L. Cao, "Negative-GSP: An Efficient Method for Mining Negative Sequential Patterns," In: *The 8th Australian Data Mining Conference. Data Mining and Analytics*, *AusDM2009,* 2009, vol. 101, pp. 63-67.

[9] Y.C. Zhao, H.F. Zhang, L.B. Cao, C.Q. Zhang, and H. Bohlscheid, "Efficient Mining of Event-Oriented Negative Sequential Rules," In: *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, IEEE Computer Society Press, 2008, pp. 336-342.

[10] X.J. Dong, Z.G. Zheng, L.B. Cao, Y.C. Zhao, C.Q. Zhang, J.J. Li, W. Wei, and Y.M. Ou E-NSP, "Efficient negative sequential pattern mining based on identified positive patterns without database rescanning," In: *International Conference on Information and Knowledge Management, Proceedings*, 2011, pp. 825-830.

[11] N.P. Lin, H.J. Chen, and W.H. Hao, "Mining negative sequential patterns," In: *Proceedings of the 6th WSEAS International Conference on Applied Computer Science*, Hangzhou, China, 2007. pp. 654-658.

[12] W.M. Ouyang, and Q.H. Huang, "Mining negative sequential patterns in transaction databases," In: *Proceedings of 2007 International Conference on Machine Learning and Cybernetics*, Hong Kong, China, 2007, pp. 830-834.

[13] Z. Zheng, Y. Zhao, Z. Zuo, and L. Cao, "An efficient ga-based algorithm for mining negative sequential patterns," In: *PAKDD' 10*, 2010, vol. 6118, pp. 262 -273.

[14] W.M. Ouyang, Q.H. Huang, and S.H. Luo, "Mining Positive and Negative Fuzzy Sequential Patterns in Large Transaction Databases", In: *5th International Conference on Fuzzy Systems and Knowledge Discovery*, 2008, pp. 18-23.

[15] W.M. Ouyang, and Q.H. Huang, "Mining positive and negative fuzzy multiple level sequential patterns in large transaction databases," In: *Proceedings of the 2009 WRI Global Congress on Intelligent Systems,* GCIS, 2009, vol. 1, pp. 500-504.

[16] W.M. Ouyang, and Q.H. Huang, "Mining positive and negative sequential patterns with multiple minimum supports in large transaction databases," In: *Proceedings - 2010 2nd WRI Global Congress on Intelligent Systems,* GCIS, 2010, vol. 2, pp. 190-193.

[17] V. Rastogi and V.K. Khare, "Apriori Based: Mining Positive and Negative Frequent Sequential Patterns," *International Journal of Latest Trends in Engineering and Technology (IJLTET)*, vol. 1, no. 3, pp. 24 -33, 2012.

[18] V.K. Khare and V. Rastogi, "Mining Positive and Negative Sequential Pattern in Incremental Transaction Databases," *International Journal of Computer Applications*, vol. 71, no.1, pp. 18-22, 2013.

[19] N.P. Lin, H.J. Chen, W.H. Hao, H.E. Chueh, and C.I. Chang, "Mining Negative Fuzzy Sequential Patterns," In: *Proceedings of the 7th WSEAS International Conference on Simulation, Modelling and Optimization*, Beijing, China, September 15-17, 2007, pp. 52-57.

[20] N.P. Lin, H.J. Chen, W.H. Hao, H.E. Chueh, and C.I. Chang, "Mining Strong Positive and Negative Sequential Patterns," WSEAS *Transactions on Computers*, vol. 7, no. 3. pp. 119-124, 2008.

Received: September 16, 2014 Revised: December 23, 2014 Accepted: December 31, 2014