Negative-GSP: An Efficient Method for Mining Negative Sequential Patterns

Zhigang Zheng **Yanchang Zhao** **Ziye Zuo** **Longbing Cao**

Data Sciences & Knowledge Discovery Research Lab
Centre for Quantum Computation and Intelligent Systems
Faculty of Engineering & IT, University of Technology, Sydney, Australia
Email: zgzheng, yczhao, zyzuo, lbcao @it.uts.edu.au

Abstract

Different from traditional positive sequential pattern mining, negative sequential pattern mining considers both positive and negative relationships between items. Negative sequential pattern mining doesn't necessarily follow the Apriori principle, and the searching space is much larger than positive pattern mining. Giving definitions and some constraints of negative sequential patterns, this paper proposes a new method for mining negative sequential patterns, called Negative-GSP. Negative-GSP can find negative sequential patterns effectively and efficiently by joining and pruning, and extensive experimental results show the efficiency of the method.

Keywords: Negative Sequential Pattern, Sequence Mining, Data Mining

1 Introduction

The concept of discovering sequential patterns was firstly introduced in 1995(Agrawal et al. 1995), aiming at discovering frequent subsequences as patterns in a sequence database, given a user-specified minimum support threshold. Some popular algorithms on sequential pattern mining include AprioriAll(Agrawal et al. 1995), GSP(Generalized Sequential Patterns)(Srikant et al. 1998) and PrefixSpan(Pei et al. 2004). GSP and AprioriAll are both Apriori-like methods based on breadth-first search, while PrefixSpan is based on depth-first search. Some other methods such as SPADE(Sequential PAttern Discovery using Equivalence classes)(Zaki 2001) and SPAM(Sequential PAttern Mining)(Jay et al. 2002) are also widely used in researches.

Different from traditional positive sequential patterns, negative sequential patterns focus on negative relationship between itemsets, in which case, absent items are taken into consideration. We give a simple example to illustrate the differences: Suppose p_1=$<a\ b\ c\ d\ f>$ is a positive sequential pattern; p_2=$<a\ b\ \neg c\ e\ f>$ is a negative sequential pattern; and each item a, b, c, etc, stands for a medical item code in the customer claim database of a private health care insurance company. By getting pattern p_1, we can tell that an insurant usually claimed for a, b, c, d and f in a row; but with pattern p_2, we are also able to find that given an insurant claim for medical items a and b, and the customer does NOT claim c, he/she would claim item e instead of d later. This kind of patterns can't be described or discovered by positive sequential patterns like p_1.

However, while we tried to apply traditional frequent pattern mining algorithm to the negative patterns, two problems stand in the way:

1. Apriori principle doesn't apply to negative sequential patterns. Apriori principle can be simply described as: a sequence can not be frequent if any of its sub-sequences is not. The Apriori principle is widely adopted to reduce the candidate subsequences in positive patterns(Agrawal et al. 1995, Srikant et al. 1998), but it is not necessarily true with patterns containing negative items. Take c_1=$<b\ \neg c>$, c_2=$<b \neg c\ a>$ as two candidate patterns and s=$<b\ d\ a\ c>$ as sequence data. We can see that s supports c_2 but doesn't support c_1, which is to say, the pattern c_2 may have greater support than c_1 although c_2 has one more element. We are going to discuss this problem thoroughly in Section 3.

2. Due to the vast candidate space, how can we find frequent patterns efficiently and effectively? Take a 3-length sequence $<a\ b\ c>$ for instance, it can only support positive candidate patterns $<a>$, $<b>$, $<c>$, $<a\ b>$, $<a\ c>$, $<b\ c>$ and $<a\ b\ c>$. But in the negative case, sequence $<a\ b\ c>$ is not only matched with the above positive patterns, but also can be matched with a large bunch of negative candidates, such as $<a\ \neg a>$,$<b\ \neg a>$,$<b\ \neg b>$, $<a\ \neg a\ c>$, $<a\ \neg c\ c>$ etc, which makes the searching space much larger.

In this paper, we propose a new method, Negative-GSP, for mining negative sequential patterns based on the GSP algorithm. We also improve the joining and pruning steps for negative sequences to ensure search space integrality and reduce the number of negative candidates as well. Our experiments show that our method can find negative sequential patterns effectively.

2 Related Work

Before negative sequential pattern mining was proposed, a couple of methods have been designed to find negative association rules(Savasere et al. 1998, Wu et al. 2004, Antonie et al. 2004) and negative sequential rules (Zhao et al. 2008). Most early researches on sequential patterns focused on positive relationships, and in recent years, a few researches start to focus on negative sequential pattern mining. The following are some researches pressed in recent years.

Ouyang & Huang(Ouyang et al. 2007) extended traditional sequential pattern definition *(a,b)* to include negative element like *(¬a,b)*, *(a,¬b)* and *(¬a,¬b)*. They put forward an algorithm which finds both frequent and infrequent sequences and then obtains negative sequential patterns from infrequent sequences. A drawback of the algorithm is that a large amount of space is required in order to find both frequent and infrequent sequences.

Nancy et al.(Nance et al. 2007) designed an algorithm named PNSPM (Positive and Negative sequential pattern mining) for mining negative sequential patterns. They applied the Apriori principle to prune redundant candidates, and extracted meaningful negative sequences using the interestingness measure. According to their pattern definition, all elements must be positive except for the last one.

Ouyang et al.(Ouyang et al. 2008) presented the definitions of three types of negative fuzzy sequential patterns: $(a,\neg b)$, $(\neg a,b)$ and $(\neg a,\neg b)$, and then described a method for mining native fuzzy sequential patterns from quantitative valued transactions.

The GSP algorithm(Srikant et al. 1998) is a classical and widely recognized algorithm for sequential pattern mining. GSP makes multiple passes over the dataset to generate frequent sequential patterns. The first pass starts from calculating the support of each single item. At the end of the first pass, all of the 1-item patterns are obtained, which are then used as seeds to generate new candidates for the next pass. Each new candidate has one more item than its seed. The candidates are then pruned to remove infrequent ones. After pruning, the supports of the new candidates are counted by another pass over the dataset and frequent patterns become the seeds for the next pass. The algorithm terminates when there are no frequent patterns at the end of a pass, or when no candidates are generated.

Sue-Chen et al.(Sue-Chen et al. 2008) proposed an algorithm called PNSP (Positive and Negative sequential pattern mining). They presented more comprehensive definitions about negative sequential patterns and extended GSP algorithm to deal with mining negative sequential patterns. Two concepts called *n-cover* and *n-contain* are employed to guide the method. It was claimed that if a *n-cover* value of a candidate is less than the *min_support*, any of its super-sequence is not going to be frequent so the searching of candidate is ended. However, we found out it may cause a loss of some candidates, and so we proposed new measurement to generate and prune candidates.

3 Problem Statement

3.1 Negative Sequence

Definition 1: Sequence

A *sequence* s is an ordered list of elements, $s=<e_1\ e_2\ ...\ e_n>$, where each e_i, $1 \leq i \leq n$, is an element. An *element* $e_i (1 \leq i \leq k)$ includes one or more items. For example, sequence $<a\ b\ (c,d)\ e>$ include 4 elements and (c,d) is an element which includes two items.

The *length* of a sequence is the number of items in the sequence. A sequence with k items is called a *k-sequence* or *k-item sequence*.

Definition 2: Positive/Negative Sequence

A sequence $s=<e_1\ e_2\ ...\ e_n>$ is a positive sequence, when each element e_i, $1 \leq i \leq n$ is a positive element. A sequence $s=<e_1\ e_2\ ...\ e_n>$ is a negative sequence, when $\exists i$, $1 \leq i \leq n$, e_i is a negative element representing the absence of an element. For example, $\neg c$ and $\neg(c,d)$ are negative elements, so $<a\ b\ \neg c\ f>$ and $<a\ b\ \neg(c,d)\ f>$ are both negative sequences.

Definition 3: Subsequence

A sequence $s_r=<r_1\ r_2\ ...\ r_m>$ is a subsequence of another sequence $s_p=<p_1\ p_2\ ...\ p_n>$, if there exists $1 \leq i_1 \leq i_2 \leq ... \leq i_k \leq n$, $r_1 \subseteq p_{i_1}$, $r_2 \subseteq p_{i_2}$, ..., $r_k \subseteq p_{i_k}$.

Definition 4: Maximum Positive Subsequence

A sequence s_r is a maximum positive subsequence of another sequence s_p, if s_r is a subsequence of s_p, and s_r includes all positive elements of s_p. For example, $<a\ b\ f>$ is maximum positive subsequence of $<a\ b\ \neg c\ f>$ and $<a\ b\ \neg(c,d)\ f>$.

Table 1: Examples of pattern matching

Pattern	base-match	match	Sequence
$<b\ \neg c\ a>$	$\surd$	$\surd$	$<b\ d\ a>$
$<b\ \neg c\ a>$	$\surd$	$\surd$	$<b\ d\ a\ c>$
$<b\ \neg c\ a>$	$\surd$	$\times$	$<b\ d\ c\ a>$

Definition 5: Negative Sequential Pattern

If the support of a negative sequence is greater than a pre-defined support threshold *min_sup*, and it meets the following constraints, then we call it a negative sequential pattern.

1) Items in a single element should be all positive or all negative. For example, $<a\ (a,\neg b)\ c>$ is not allowed because item a and item $\neg b$ are in a same element;

2) Two or more continuous negative elements are not accepted in the negative sequence. This constraint is also used by other researchers(Sue-Chen et al. 2008).

3) For each negative item in a negative pattern, its corresponding positive item is required to be frequent. For example, if $<\neg c>$ is a negative item, its corresponding positive item $<c>$ is required to be frequent.

3.2 Negative Pattern Matching

In order to calculate the support of a negative sequential pattern against the sequence data in database, we need clarify pattern-sequence matching method and criterion. We need to define which patterns can a sequence support.

Definition 6: Negative Base-matching

A negative sequence $s_n=<e_1\ e_2\ ...\ e_k>$ *base-matches* a data sequence $s=<d_1\ d_2\ ...\ d_m>$, iff, for every negative element e_i, there exist integers p, q, r $(p<q<r)$ such that the two conditions hold:

(1) s contains the maximum positive subsequence of s_n,

(2) $\exists e_{i-1} \subseteq d_p$ *and* $e_{i+1} \subseteq d_r$, and $\exists d_q$, $e_i \not\subseteq d_q$

That is, each positive element in s_n matches with the same elements in s with same order, while each negative element of s_n can find a match element in s at the corresponding position.

For example, $s_n=<b\ \neg c\ a>$, it *base-matches* $<b\ d\ a>$, and also base-matches $<b\ d\ c\ a>$, since the element d, which matches $\neg c$, can be found between the element b and a.

If a sequence base-match a pattern, then we should count it in $base_s upport$ value. We use base-match to find seed sequences, which will ensure the integrity of result patterns.

Definition 7: Negative Matching

A negative sequence $s_n=<e_1\ e_2\ ...\ e_k>$ *matches* a data sequence $s=<d_1\ d_2\ ...\ d_m>$, iff, for every negative element e_i, there exist integers p, q, r $(p<q<r)$ such that the two conditions hold:

(1) s contains the maximum positive subsequence of s_n,

(2) $\exists e_{i-1} \subseteq d_p$ *and* $e_{i+1} \subseteq d_r$, and $\forall d_q$, $e_i \not\subseteq d_q$

For example, $s_n=<b\ \neg c\ a>$ matches $<b\ d\ a\ c>$, but does not match $<b\ d\ c\ a>$, since the negative element c appears between the element b and a in s_n.

The above match definition is same as n-cover concept in(Sue-Chen et al. 2008). (Sue-Chen et al. 2008) also defined n-contain, which is a more restricted match criterion.

A pattern-sequence matching example is given in Table 1. Based on the matching method we have, Apriori-property is not suitable for this case, as we

Table 2: Examples for explain Apriori-Principle

Pattern	match	Sequence
s_{n1}=<b ¬c a>	×	<b f d c a>
s_{n2}=<b ¬c d a>	√	<b f d c a>

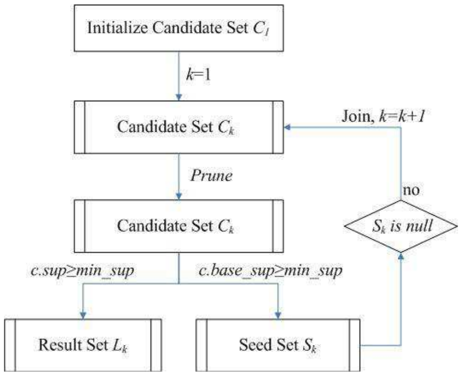

Figure 1: Process Flow of Negative-GSP

pointed out in Section 1. Table 2 gives out a straightforward example. s_{n1}=<b ¬c a> and s_{n2}=<b ¬c d a> are two candidate patterns and <b f d c a> is a sequence in the dataset. The table clearly shows that s_{n1} does not match s, while s_{n2} matches s even if s_{n1} is a sub-sequence of s_{n2}.

Property: Negative Frequent Pattern Property

Based on the above definitions, apparently we have a property: if a negative sequence s is frequent, all its positive subsequence must be frequent; or if the maximum positive subsequence of a sequence s is not frequent, s can not be frequent.

4 Negative-GSP Algorithm

4.1 The Idea

Assume in a sequence data set D_s ={d_1, d_2, d_3, ..., d_n}, where d_i *(1≤i≤n)* is a sequence, and our objective is to find frequent negative sequential patterns in D_s, with a minimum support threshold *min_sup*. The basic process flow as Fig. 1.

First, we utilize the GSP algorithm to generate all positive sequential patterns. Assume L_{pos}={$L_{pos,1}$, $L_{pos,2}$, ..., $L_{pos,l}$}, where $L_{pos,i}$ represents the i-item frequent positive sequential pattern set.

Next, we begin to generate negative sequential patterns based on L_{pos}. We transform all 1-item positive sequential patterns $L_{pos,1}$ to their corresponding 1-item negative sequences, which are taken as the initial 1-item candidates set $C_{neg,1}$. Then we also need prune unnecessary candidates from candidates set $C_{neg,1}$ before calculating the support of every candidates by passing over D_s, which will be discussed in Section 4.3.

Because Apriori Principle doesn't work for negative sequence, the 1-item candidates need to be divided into two classes: one is valid to join with other candidates to generate 2-item negative candidates, and the other is invalid. To verify whether they are valid in the next pass, we use an *base-match* method to verify it (refer to Section 4.4). After this step, we get a processed 1-item seed set $S_{neg,1}$, which is then used as seed to generate a 2-item candidate set $C_{neg,2}$. The candidates with support higher than *min_sup* are outputted to 1-item frequent patterns $L_{neg,1}$.

Based on the k-item seed set $S_{neg,k}$, we join them with the joining method presented in Section 4.2 to produce a (k+1)-item candidates set $C_{neg,k+1}$. The new candidates may include many invalid candidates, so pruning invalid candidates is necessary and helpful for further search. Our idea is to verify whether the maximum positive subsequence of the candidate is frequent. Invalid candidates are pruned, and valid ones are kept in the seed set. Then, by passing over the data set D_s, we get the supports of all (k+1)-item candidates. Again, the (k+1)-item frequent candidates with support higher than *min_sup* are outputted to $L_{neg,k+1}$ as (k+1)-item frequent patterns.

After the above operation, we get the (k+1)-item frequent result $L_{neg,k+1}$ and (k+1)-item seed set $S_{neg,k+1}$ for next pass. Longer patterns are generated by repeating the above process until the candidate set becomes empty. Each iteration will generate and output frequent negative sequential patterns into L_{neg}={$L_{neg,1}$, $L_{neg,2}$, ..., $L_{neg,l}$}, which is the final set of negative sequential patterns.

4.2 Joining to Generate Candidates

The (k+1)-item candidates are generated by joining all k-item seed sequences. Given two k-item seed sequences c_i=<e_{i_1} e_{i_2} e_{i_3} ... $e_{i_{k-1}}$ e_{i_k}> and c_j=<e_{j_1} e_{j_2} e_{j_3} ... $e_{j_{k-1}}$ e_{j_k}>, assume c_i'=<e_{i_1} e_{i_2} e_{i_3} ... $e_{i_{k-1}}$>, c_j'=<e_{j_2} e_{j_3} ... $e_{j_{k-1}}$ e_{j_k}>. If c_i'=c_j', c_i and c_j are joined to get a (k+1)-item candidate: <e_{j_1} e_{i_1} e_{i_2} ... $e_{i_{k-1}}$ e_{i_k}>.

The above joining method is similar to but different from the joining method of GSP. The GSP algorithm gets all (k+1)-item candidates by joining k-item frequent sequential patterns since positive sequences obey the Apriori principle. However, when we mine negative sequential patterns, we need to join not only k-item frequent patterns but also some infrequent k-item sequences, as demonstrated by Section 4.3.

Another point is that, while performing the joining operation, we ignore the joining of positive patterns with themselves, since they have been done in the positive sequential pattern mining at the first step of our algorithm.

4.3 Pruning Unnecessary Candidates

Since the candidates of negative sequential patterns generated during joining step are much more than positive sequential patterns, it is necessary to design an effective pruning method for negative sequential pattern mining. This step prunes some unnecessary candidates from candidates set.

With the GSP algorithm, while pruning in k-item candidates, it prunes all the candidates whose (k-1)-subsequences are not frequent. However, the above pruning method does not work for negative sequential pattern mining in the following ways.

Firstly, given a candidate C=<a b ¬c d ¬e>. Note that we don't allow two continuous negative elements in a sequential pattern, so C'=<a b ¬c ¬e> is an invalid negative sequential pattern. As a result, C will be pruned. However, in fact, the candidate C may be a valid candidate of negative sequential pattern. Secondly, C may be frequent even when its subsequence is not frequent. So we can't prune C simply even its subsequence is infrequent.

Our pruning method is described as follows. For a candidate C=<e_1 e_2 e_3 ... e_n>, suppose C'=<e_i e_j ... e_k> is the maximum positive subsequence of C. If C' is not frequent, C must be infrequent and should be pruned. This method is simple but effective to prune invalid candidates without cutting off possible valid candidates by mistake.

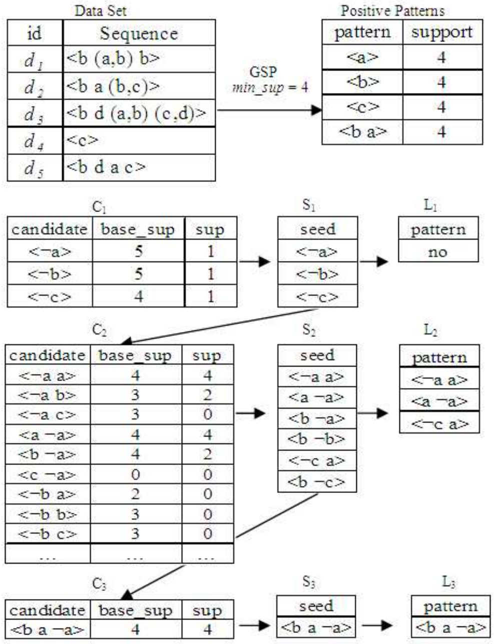

Figure 2: An example

4.4 Generating Seed Set for Next Pass

This step keeps all necessary k-item seed sequences in seed set for next pass, and then (k+1)-item candidates are generated by joining both frequent and infrequent k-item seed sequences.

Given an infrequent 2-item sequence $<b\ \neg c>$, 3-item candidate $<b\ \neg c\ d>$ may still be frequent. So we need count $<b\ \neg c>$ as seed sequence for joining and generate 3-item candidate $<b\ \neg c\ d>$.

It is proposed that the k-item sequences will be regarded as k-item seed sequences if their *base-supports* are greater than *min_sup*. So we check each candidate's *base-support* value to see whether it is greater than *min_sup*. If not, it means that the sequence is impossible to generate (k+1)-item frequent pattern and we don't need count it in the seed set again.

4.5 Algorithm Description

Our proposed algorithm for negative sequential pattern mining is given below.

Step 1: Find all positive sequential patterns by the traditional GSP algorithm(Srikant et al. 1998).

Step 2: Transform 1-item positive patterns to 1-item negative patterns as candidates set, and then get 1-item seed set and 1-item patterns.

Step 3: Use all (k-1)-item seeds, perform the joining operation with each other and get k-item candidates and also join (k-1)-item positive patterns with (k-1)-item seed sequences.

Step 4: Prune unnecessary candidates to get a smaller candidate set.

Step 5: Count all candidates' supports and base-supports.

Step 6: If the base-support is greater than *min_sup*, then the candidate is added to k-item seed set. If the support is greater than *min_sup*, then the candidate is frequent and is outputted as a k-item pattern.

Step 7: If k-item seed set is not empty, increase k by one and loop back to Step 3 until the next candidate set is empty.

The procedure is illustrated with an example in Fig 2.

Table 3: Synthetic datasets

Parameters	*DS1*	*DS2*
Number of sequences (*DB*)	10k	100k
Number of items (*N*)	1k	10k
Average number of elements per sequence (*C*)	8	10
Average number of items per element (*T*)	8	2.5
Average length of maximal potentially large sequences (*S*)	4	4
Average size of itemsets in maximal potentially large sequences (*I*)	8	2.5

Negative-GSP Algorithm

Input: Dataset Ds, min_sup
Output: Negative patterns L

```
/*  S: Seed set;
    C: Candidate set;
    L: Result set; */

k = 1;
C_k = initialize();
S_k = C_k;
while ( S_k.size() > 0 ){
    C_{k+1} = Join(S_k);
    C_{k+1} = Prune(C_{k+1});
    for ( each c in C_{k+1} ){
        if ( c.basesupport > min_sup ) S_{k+1}.add(c);
        if ( c.support > min_sup ) L_{k+1}.add(c);
    }
    L.add(L_{k+1});
    k = k + 1;
}
return L;
```

5 Experiments and Results

5.1 Datasets

Two synthetic datasets generated by IBM data generator(Agrawal et al. 1995) are used to test our algorithm in the experiments.

$Dataset1(DS1)$ is C8.T8.S4.I8.DB10k.N1k, which contains 10k sequences and the number of items is 1000, the average number of elements in a sequence is 8, the average number of items in an element is 8, average length of maximal pattern consists of 4 elements and each element is composed of 8 items averagely.

$Dataset2(DS2)$ is C10.T2.5.S4.I2.5.DB100k.N10k, which contains 100k sequences and the number of items is 10k, the average number of elements in a sequence is 10, the average number of items in an element is 2.5, average length of maximal pattern consists of 4 elements and each element is composed of 2.5 items averagely.

5.2 Experiments Results

We compared the runtime of negative sequential pattern mining with different support thresholds (see Fig 3), and also compared counts of patterns with different support thresholds (see Fig 4). Negative pattern mining spends much more runtime than positive pattern mining because the candidates counts are not of the same magnitude, especially when the support threshold is set very low.

5.3 Comparison with PNSP Algorithm

We compared our method with PNSP algorithm(Sue-Chen et al. 2008). The results of runtime (see Fig 5)

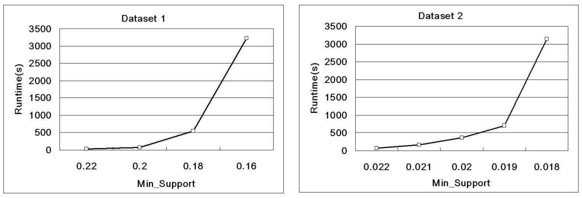

Figure 3: Runtime on DS1 and DS2

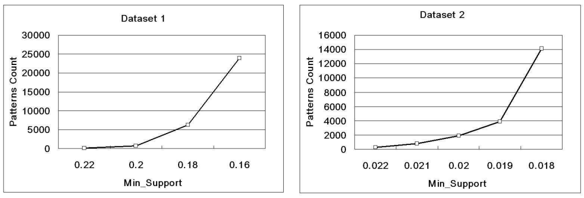

Figure 4: Patterns counts on DS1 and DS2

show that our method outperforms PNSP. The reason is that PNSP generates more unnecessary candidates. Especially when the number of negative frequent items increased, its negative candidates may increase sharply. For our algorithm, when there are a lot of negative candidates, it will cost much running time in the joining process for new candidates. So when *min_sup* is very low, Negative-GSP can't get very good performance as when *min_sup* is high.

6 Conclusions and Future Work

In this paper, we presented the definitions for negative sequential patterns, and proposed a method for negative sequential pattern mining based on the GSP algorithm. We also designed effective pruning method to reduce the number of candidates. The efficiency and effectiveness of our algorithm is shown in our experiments on two synthetic datasets.

In our future research, we will focus on selecting interesting rules from the discovered negative sequential patterns. How to find interesting and interpretable rules from a lot of negative sequential patterns is valuable in business applications. Another research topic is to find more effective pruning method that can reduce candidates more effectively and avoid unnecessary candidates.

Acknowledgements

This work was supported by the Australia Research Council (ARC) Linkage Project LP0775041 and Discovery Projects DP0667060 & DP0773412, and by the Early Career Researcher Grant from University of Technology, Sydney, Australia. The author of this paper acknowledge the financial support of the Capital Markets CRC.

References

Agrawal, R. & Srikant, R. (1995), Mining Sequential Patterns. In: Yu, P.S., Chen, A.L.P. (eds.): 11th International Conference on Data Engineering. I E E E, Computer Soc Press, Taipei, Taiwan, pp. 3–14

Antonie, M.L. & Zaiane, O.R. (2004),Mining positive and negative association rules: An approach for confined rules. Proceedings of the 15th European Conference on Machine Learning/8th European Conference on Principles and Practice of Knowledge Discovery in Databases. Springer-Verlag Berlin, Pisa, ITALY,pp. 27–38

Jay, A., Jason, F., Johannes, G. & Tomi, Y. (2002), Sequential PAttern mining using a bitmap representation. Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, Edmonton, Alberta, Canada

Nancy, P.L., Hung-Jen, C. & Wei-Hua, H. (2007), Mining negative sequential patterns. Proceedings of the 6th Conference on WSEAS International Conference on Applied Computer Science - Volume 6. World Scientific and Engineering Academy and Society (WSEAS), Hangzhou, China

Ouyang, W., Huang, Q. & Luo, S. (2008), Mining Positive and Negative Fuzzy Sequential Patterns in Large Transaction Databases. Fuzzy Systems and Knowledge Discovery, 2008. FSKD '08. Fifth International Conference on, Vol. 5, pp. 18–23

Ouyang, W.M. & Huang, Q.H. (2007), Mining Negative Sequential Patterns in Transaction Databases. Machine Learning and Cybernetics, 2007 International Conference on, Vol. 2, pp. 830–834

Pei, J., Han, J., Mortazavi-Asl, B., Jianyong, W., Pinto, H., Qiming, C., Dayal, U. & Mei-Chun, H. (2004), Mining sequential patterns by pattern-growth: the PrefixSpan approach. Knowledge and Data Engineering, IEEE Transactions on 16, pp. 1424–1440

Savasere, A., Omiecinski, E. & Navathe, S. (1998), Mining for strong negative associations in a large database of customer transactions. 14th International Conference on Data Engineering, Proceedings, pp. 494–502

Srikant, R. & A., R. (1998), Mining sequential patterns: Generalisations and performance improvements. Proceedings of the Fifth International Conference on Extending Database Technology (EDBT)

Sue-Chen, H., Ming-Yen, L. & Chien-Liang, C. (2008), Mining Negative Sequential Patterns for E-commerce Recommendations. Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference. IEEE Computer Society

Wu, X.D., Zhang, C.Q. & Zhang, S.C. (2004), Efficient mining of both positive and negative association rules. ACM Trans. Inf. Syst. 22, pp. 381–405

Zaki, M.J. (2001), SPADE: An efficient algorithm for mining frequent sequences. Machine Learning 42, pp. 31–60

Zhao, Y., Zhang, H., Cao, L., Zhang, C. & Bohlscheid, H. (2008), Efficient Mining of Event-Oriented Negative Sequential Rules. Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08. IEEE/WIC/ACM International Conference on, Vol. 1, pp. 336–342

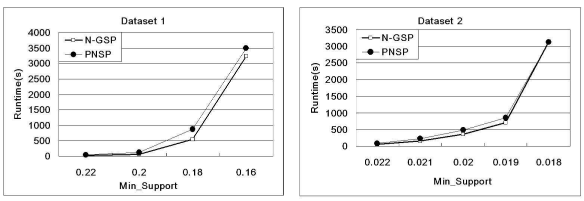

Figure 5: Comparison with PNSP Algorithm