Disjunctive Sequential Patterns on Single Data Sequence and its Anti-Monotonicity

Kazuhiro SHIMIZU[1] and Takao MIURA[1]

Dept.of Elect.& Elect. Engr., HOSEI University
3-7-2 KajinoCho, Koganei, Tokyo, 184–8584 Japan

Abstract. In this work, we proposes a novel method for mining frequent disjunctive patterns on single data sequence. For this purpose, we introduce a sophisticated measure that satisfies *anti-monotonicity*, by which we can discuss efficient mining algorithm based on APRIORI. We discuss some experimental results.
Keywords: Disjunctive Sequence Pattern, Anti-Monotonicity, Single Data Sequence

1 Motivation

Recently there have been much investigation discussed to text data analysis using data mining techniques proposed so far. This is because generally text data has been analyzed qualitatively but not quantitatively, which means it is hard to apply several quantitative techniques such as statistical tests and data mining approach[6, 9]. However, much attention has been focused on quantitative analysis of several features of text thus new approach, called *text mining*, has been proposed to extend traditional methodologies.

Main ideas in text mining approach come from *frequency* and *co-occurrence*, the former means important words arise many times while the latter says related words occur at the same time[12, 14]. For example, when we analyze purchase logs in some bookstore, we could obtain co-occurrence of *"Snakes and Earrings"* and *"The Back One Wants to Kick"* and arrange the books on shelf or promoting the campaign[1].

The idea can be extended to many activities. Here we discuss em sequence data that means an ordered list of information along with time or any other and we extract patterns from the list as common behavior. In our case, we could obtain correlation that says those who get *Snakes and Earrings* would also get *The Back One Wants to Kick* in a month, and we would arrange some promotion behavior such as direct-mail or any other campaign *in the month*. Also we could apply the techniques for analyzing Web access patterns, medical-care and DNA sequence[8, 10, 11].

Among others, text mining has been paid much attention recently where *text* describes some context consisting of sequence of words. The main idea is similar

[1] The both books win the 130-th *Akutagawa* award in 2003. The authors, Ms. Hitomi Kanehara and Ms. Risa Wataya, are in 20's and have gotten into the news in Japan

to data mining, i.e., *frequency* and *co-occurrence*. We extract frequent patterns (lists of words, phrase) and we summarize (abstract) and label important events from them.

Clearly it is hard to extract all the frequent patterns from very long lists of words, since the search-space is so huge that we face to combinatorial explosion to the problems. For instance, we might have some sentences:

(1) I met a mother of my friend's brother. Next day, I met a mother of my brother's friend.
(2) I saw a red and white flag[2]. Next day, I saw a white and red flag.

Clearly, in (1), we have two distinct mothers while, in (2), we see a same flag. To extract frequent patterns, we should count separately in (1) but count double in (2). In text mining, we want to extract a pattern *[red, white] flag* that means a pattern contains both a *red white flag* and a *white red flag* by ignoring permutation of *red* and em white. This is called a *disjunctive* pattern or a *permuted* pattern.

We might think about regular expression by adding Kleene closure but we avoid the extension because of vast amount of complexity in this work.

When we examine disjunctive patterns, we might generate huge number of candidate patterns such as "*Snakes and Earrings The Back One Wants to Kick*". To obtain smaller search space, we need to scan a database many times thus we should have heavy I/O access to the data on hard disks. Sampling or any specialized data storage techniques might be helpful but the performance efficiency depends heavily on data distribution.

There have been important research results obtained based on APRIORI[6, 9]. Generally we can avoid vast range of search but not enough to sequence data. In general APRIORI based approach, when q is a *sub-pattern* of a pattern p, any sequence data that matches p should also match q. This property (called *anti-monotonicity* of patterns) provides us with the reduction of search space.

However, in a case of text mining, this is not true and we can't apply APRIORI technique any more. For example, in a text `"aabbba"`, a is a sub-pattern of ab but we see a pattern ab matches 6 times, both a and b 3 times respectively and $[ab]$ 9 times. In other words, we can't have a naive counting method any more.

In this investigation, given an integer m and single long sequence data S, we discuss how to extract p where a new count $\mathcal{M}_S(p)$ is more than m. Here the *true problem* is the counting scheme $\mathcal{M}$. We extend some approach proposed[16] and discuss a new framework for disjunctive patterns within APRIORI.

In section 2 we formalize our problem, and, in section 3, we introduce a new measure for counting satisfying anti-monotonicity. Section 4 contains some experimental results.

[2] The national flag of Japan consists of red and white parts.

2 Text Mining From Single Sequence Data

In this work, we consider a word as a unit, called *item*. Any element in an itemset $I = \{i_1, .., i_L\}, L > 0$ is called *alphabet*, and a *sequence data* (or, text) $S = s_1...s_m, m > 0$ is an ordered list of items, m is called a *length* of S, and S is called m-sequence. Note an item may appear many times in S.

A *disjunctive pattern* (or just a pattern) p is a form of $t_1...t_n, n > 0$ where each t_i is an alphabet a or a disjunction $[a_1a_2...a_m], m > 0$, each a_j is a distinct alphabet.

Given two patterns $p = t_1...t_n, n > 0$ and $q = v_1v_2...v_m, m \leq n$, we say q is a *sub-pattern* of p, denoted by $q \sqsubseteq p$, if there exist $1 \leq j_1 < ... < j_m \leq n$ such that each v_k corresponds to t_{j_k} (denoted by $v_k \sqsubseteq t_{j_k}$ if no confusion arises) satisfying:

If v_k is an alphabet a, $t_{j_k} = a$ or t_{j_k} is a disjunction containing a
If v_k is a disjunction $[a_1a_2...a_m]$, we have both $t_{j_k} = [b_1b_2...b_l]$ and $\{a_1, .., a_m\} \subseteq \{b_1, .., b_l\}$

EXAMPLE 1 ac is a sub-pattern of $abcd$. Similarly $[ac]$ is a sub-pattern of $[abcd]$, bd is a sub-pattern of $[ab]b[cd]de$, b is a sub-pattern of $[ab]$, and "ac" is a sub-pattern of $[ab][cd]$.

However, ab is not a sub-pattern of $[ab]$, nor $[ab]$ is a sub-pattern of ab.

We say a pattern $p = t_1t_2...t_n$ matches a sequence $S = c_1c_2...c_m$

if t_1 is an alphabet a_1, there exist $t_1 = a_1 = c_{i_1}, 1 \leq i_1 \leq m$ and the sub-pattern $t_2...t_n$ matches $c_{i_1+1}...c_m$,
and if t_1 is a disjunction $[a_1a_2...a_m]$, there exists a permutation $a_{j_1}...a_{j_m}$ of $a_1, ..., a_m$ that matches $c_1...c_{i_1}$, and the subpattern $t_2...t_n$ matches $c_{i_1+1}...c_m$.

EXAMPLE 2 Assume S is `aabbba`. A pattern a matches S 3 times, ab 6 times and $[ab]$ 9 times. Note we can see more frequency by $[ab]$.

Given a sequence S, a function $\mathcal{M}_S$ from patterns to non-negative integers satisfies *Anti Monotonicity* if for any patterns p, q such that $q \sqsubseteq p$, we have $\mathcal{M}_S(q) \geq \mathcal{M}_S(p)$. In the following, we assume some S and we say $\mathcal{M}$ for $\mathcal{M}_S$.

Given $\mathcal{M}$ and an integer $m > 0$ (called *minimum support*), A pattern p is called *frequent* if $\mathcal{M}(p) \geq m$. If $\mathcal{M}$ satisfies anti-monotonicity, for any q such that $\mathcal{M}(q) < m$, there is no frequent p such that $q \sqsubseteq p$. By using this property, we can reduce search space to extract frequent patterns. In fact, this is the motivation of APRIORI[1] and the algorithm is given as follows:

(1) Find frequent patterns of the minimum size
(2) Find one-size larger patterns p where all the sub patterns are frequent. (The results are called *candidate*.) Stop if there is no pattern obtained.
(3) Select all the frequent patterns from the candidates by scanning S and goto (2).

In this work, given a sequence S, we examine all the frequent patterns p. However, it is not easy to obtain $\mathcal{M}$ satisfying anti-monotonicity. For example, "the number of matching" is not suitable as $\mathcal{M}$ as shown in EXAMPLE 2.

3 Anti-monotonic Counting

In this section, we discuss new counting methods for anti-monotonicity since naive matching is not suitable.

The first idea is called *head frequency*. Given a sequence $S = s_1 s_2 ... s_r$ and a pattern p of $t_1 t_2 ... t_n$, we define a *head frequency* $H(S, p)$ as follows:

$$H(S, p) = \Sigma_{i=1}^{r} Val(S, i, p)$$

where $Val(S, i, p)$ is 1 if the followings hold, and 0 otherwise:

> Let $S(i)$ be a suffix of S from i-th position, i.e., $S(i) = s_i ... s_r$. If t_1 is an alphabet a, we have $s_i = a$ and $t_2 t_3 ... t_n$ matches $S(i+1)$. And if t_1 is a disjunction $[a_1 a_2 ... a_m]$, there exists j such that $s_i = a_j$ (for instance, $j = 1$), and $[a_2 a_3 ... a_m] t_2 ... t_n$ matches $S(i+1)$.

Intuitively $H(S, p)$ describes the number of matching of p from the heading of S or its suffix.

EXAMPLE 3 (1) Let S be a sequence `bbba`. If $p = ba$, we have $H(S, p) = 3$ while p matches S 3 times. If $p = a$, we have $H(S, p) = 1$ and the number of matching is 1. By the definition, we have $a \sqsubseteq ba$ but not $H(S, a) > H(S, ba)$.
(2) Let S be `aabbba`. If $p = ab$, we have $H(S, p) = 2$ and the number of matching is 6. If $p = ba$, we see $H(S, p) = 3$ and p matches S 3 times. If $p = [ab]$, we get $H(S, p) = 5$ and the pattern matches S 9 times. Finally if $p = a$, we have $H(S, p) = 3$ and the number of matching is 3. By the definition, $a \sqsubseteq [ab]$ holds but $H(S, a) > H(S, [ab])$ doesn't.

As shown in this example, the head frequency $H(S, p)$ doesn't satisfy anti-monotonicity. Note that this counting ignores matching appeared in the subsequent sequence. That's why we introduce a new counting $D(S, p)$, called $total frequency$, which means the minimum $H(S, q)$ for any $q \sqsubseteq p$.

$$D(S, p) = \mathrm{MIN}\{H(S, q) | q \sqsubseteq p\}$$

THEOREM 1 $D(S, p)$ satisfies anti-monotonicity.

(*Proof*) Given patterns p, q such that $q \sqsubseteq p$, we must have $D(S, p) = \mathrm{MIN}\{H(S, r) | r \sqsubseteq p\}$ and $D(S, q) = \mathrm{MIN}\{H(S, r) | r \sqsubseteq q\}$ by the definition. That means $D(S, q) \geq D(S, p)$ since $q \sqsubseteq p$. (*Q.E.D.*)

EXAMPLE 4 (1) Let S be `bbba`. If $p = ba$, we have $D(S, p) = 1$. And if $p = a$, we see $D(S, p) = 1$.
(2) Let S be `aabbba`. If $p = ab$ we have, $D(S, p) = 2$, if $p = ba$, we see $D(S, p) =$

3, if $p = [ab]$, we get $D(S,p) = 3$, and if $p = a$, we have $D(S,p) = 3$.
(3) Let S be caabbbc. If $p = ab$, we have $H(S,p) = 2$ and $D(S,p) = 2$. If $p = ac$, we have $H(S,p) = 2$ and $D(S,p) = 2$. And if $p = [ac]$, we see $H(S,p) = 3$ and $D(S,p) = 2$ while p matches S 4 times. In these cases, sub-patterns (i.e., a, c) of ac and $[ac]$ appear more interspersed in S and this is why *total frequency* is different from *head frequency* and the number of matching.

According to theorem 1, to obtain total frequency of p of length n, it is enough to examine head frequency of all the sub-patterns. And, as the next theorem says, it is enough to calculate the ones for all the suffixes of p in ascending order of length. Note there are only n suffixes of p thus we can reduces search space dramatically.

THEOREM 2 For a sequence S and a pattern p, $D(S,p) = \mathrm{MIN}\{H(S,p), D(S,p(2))\}$.

(*Proof*) Let S be a sequence $s_1 s_2 ... s_m$, p a pattern $t_1 t_2 ... t_n$ and $p(i)$ be a sub-pattern (suffix) $t_i ... t_n$. We show $D(S,p) = \mathrm{MIN}\{H(S,p(i)) | i = 1, .., n\}$.

Let $q = u_1 u_2 ... u_k$ be any sub-pattern of p. By the assumption, there exist $1 \leq j_1 < ... < j_k \leq n$ such that $u_i \sqsubseteq t_{j_i k}$, $i = 1, ..., k$.

We define the expansion q' of q at a position i as follows ($i > 0$).

q' is one of the followings
(i) $u_1 u_2 ... u_i v u_{i+1} ... u_k$, that is, v is inserted just after u_i.
(ii) $u_1 u_2 ... u'_i ... u_k$, that is, u_i is replaced by u'_i such that $u_i \sqsubseteq u'_i$.

We show $H(S,q) \leq H(S,q')$, that is, we show that $Val(S,i,q') = 1$ means $Val(S,i,q) = 1$. If q' is an expansion of (i), we have $Val(S,i,q) = 1$ by ignoring v at matching step. And, in the case of (ii), we have $Val(S,i,q) = 1$ by ignoring $u'_i - u_i$ part at matching step.

Because t_{j_1} can be obtained by expansions of q, we must have $H(S,q) \geq H(S,t_{j_1})$. This means, for any sub-pattern q of p, there exists $t_{j'}$ that has smaller head frequency. (*Q.E.D.*)

4 Experimental Results

4.1 Experimental Data

As a test collection for our experiments, we discuss NTCIR-3 PATENT (IR Test Collection). The collection contains several data but we discuss only PAJ English Abstract (1995). Documents in JAPIO Japanese abstracts (1995-1999) are translated into PAJ English Abstract of 4GB in total. In this experiment, we select 1000 articles in 1995 abstracts that are kept in timestamp order. We remove all the stop-words and do stemming[4] to each articles.

Here are some examples of the information (after stemming):

```
control adjust speed thresh depth regul grain culm state oper load combin
shape initi puls power high load devic regul control thresh depth grain
culm detect thresh depth sensor
```

4.2 Preliminaries

In this investigation we extract frequent patterns base on APRIORI algorithm described previously[1]. To evaluate our results, we compare $T - freq(S, p)$ method [16] with us. The minimum support is 20 which is 2%. In this experiment we manipulate disjunctive patterns of $[ab]$, $[ab]c$, $[abc]d$, ..., that is, we discuss only patterns where one disjunction appears, and we ignore patterns with multiple disjunctions such as $[ab]c[de]$.

4.3 Results

Here is the result shown in Table 1 and Figure 1. In the table, we show the numbers of frequent patterns extracted by the two methods and the distribution in the figure. Note that the number in "(...)" means the number of extracted patterns by our method but not by $T - freq$ method.

Here are some examples extracted by our method:

```
([medicin,patient]),([prepar,medicin]),([surfac, sheet])...
([prepar,medicin]patient)...
```

And the next are obtained by $T - freq$ method:

```
(medicin,prepar),(medicin,patient),(devic,capabl)...
(provid,capabl,devic)...
```

n-pattern	our method	$T - freq$ method
$n = 1$	207(0)	207
$n = 2$	121(76)	45
$n = 3$	3(1)	2
$n = 4$	0(0)	0

Table 1. Number of Frequent Patterns from Patent Data

4.4 Discussion

By examing the table 1, we can extract more patterns in $n = 2$ and $n = 3$.

More important is that we can obtain 76 disjunctive patterns ($n = 2$) only by our method. This is because disjunctive patterns match S more times thus we can see more frequency. For example, we have $[surfac, sheet]$ with the frequency 22 and $[prepar, medicin]patient$ with the frequency 21, which are ignored by $T - freq$ method.

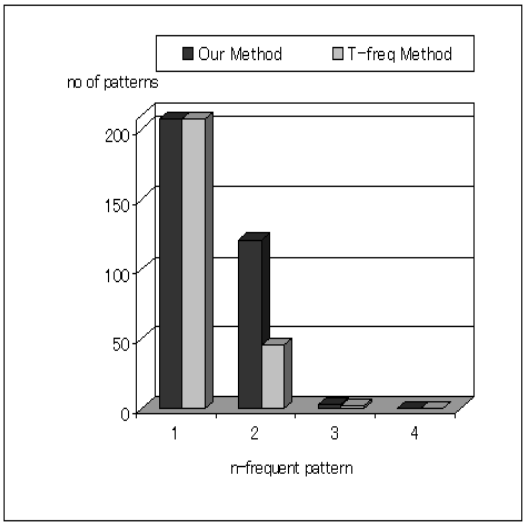

Fig. 1. Number of Frequent Patterns from Patent Data

5 Related Works

There have been several approach to obtain sequence patterns[2, 15], mainly based on APRIORI algorithm[1]. Their issue is to extract sequence patterns that appear in many sequences, given a set of sequences that consist of lists of itemsets. The problem is combinatorial but efficient algorithms have been proposed by introducing specialied data structures such as FreeSpan [5] and PrefixSpan [13], by means of lattice theory such as SPADE [17]. Some frameworks for *regular expression* have been also proposed such as SPIRIT [3] and a kind of Constraint Satisfaction Problem[7]. But in these approach the expression plays as a constraint and not targeted patterns like us. More important is that all of them discuss *multiple* sequences and that the frequency is defined in terms of the number of sequences containing the pattern. We see this is not really useful for text mining.

The problem of frequent patterns on single sequence is found in *episode* [8] where sequence is defined as a list of events. The problem for text mining is found in [16] but they don't discuss disjunctive patterns.

6 Conclusion

In this investigation, we have proposed disjunctive patterns and total frequency for the purpose of efficient pattern mining on single sequence data. The counting method satisfies anti-monotonicity thus we can obtain efficient mining algorithm based on APRIORI. By some experiments we have shown the effectiveness of the proposed method compared to traditional approach, especially we have shown

more frequent patterns extracted. Now we extend our approach to more powerful expression to get potential patterns.

Acknowledgment

We would like to acknowledge the financial support by Grant-in-Aid for Scientific Research (C)(2) (No.16500070) from Ministry of Education, Culture, Sports, Science and Technology in Japan. Also we would like to acknowledge the support of NTCIR-3 PATENT (IR Test Collection) from National Institute of Informatics (NII) in Japan.

References

1. Agrawal, R. and Srikant, R.: Fast Algorithm for Mining Association Rules, proc. VLDB, 1994, pp.487-499
2. Agrawal, R. and Srikant, R.: Mining Sequential Patterns, proc. ICDE, 1995, pp.3-14
3. Garofalakis, M., Rastogi, R. and Shim, K.: SPIRIT : Sequential Pattern Mining with Regular Expression Constraints, proc. VLDB, 1999, pp.223-234
4. Grossman,D. and Frieder,O.: Information Retrieval – Algorithms and Heuristics, Kluwer Academic Press, 1998
5. Han, J., Pei, J. Mortazavi, B., Chen, Q, Dayal, U. and Hsu, M-C.: FreeSpan: Frequent Pattern-Projected Sequential Patterns Mining, proc. KDD, 2000, pp.355-359
6. Han,J. and Kamber, M.: Data Mining : Concepts and Techniques, Morgan Kaufmann, 2000
7. Albert-Lorincz, H. and Boulicaut, J-F.: Mining Frequent Sequential Patterns under Regular Expressions: A Highly Adaptive Strategy for Pushing Constraints, proc. SIAM DM, 2003, pp.316-320
8. Mannila, H. and Toivonen, H. and Verkamo, I.: Discovery of Frequent Episodes in Event Sequences, *Data Mining and Knowledge Discovery* 1(3), 1997, pp.259-289
9. Hand, D., Mannila, H. and Smyth, P.: Principles of Data Mining, MIT Press, 2001
10. Motoyoshi, M., Miura, T., Watanabe, K. and Shioya, I.: Temporal Class Mining for Time Series Data, proc. CIKM, 2002
11. Motoyoshi, M., Miura, T. and Shioya, I.: Clustering by Regression Analysis, International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2003), pp.202-211 (Springer LNCS 2737), 2003, Prague
12. Nagao, M.: Natural Language Processing (in Japanese), Iwanami, 1996
13. Pei, J., Han, J., Mortazavi, B., Pinto H., Chen, Q, Dayal, U. and Hsu, M-C.: PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Patter Growth, proc.ICDE, 2001, pp.215-224
14. Sebastiani, F.: Machine learning in automated text categorization, ACM Comp.Survey 34-1, 2002, pp.1–47
15. Srikant, R and Agrawal, R.: Mining Sequential Patterns: Generalizations and Performance Improvements, proc.EDBT, 1996, pp.412-421
16. Takano,Y., Iwanuma, K. and Nabeshima, H.: A Frequency Measure of Sequential Patterns on a Single Very-Large Data Sequence and its Anti-Monotonicity, in Japanese, proc.*FIT*, 2004, pp.115-118
17. Zaki, M.J.: Efficient Enumeration of Frequent Sequences, proc.CIKM, 1998, pp.68-75