MINING FOR STABLE PATTERNS:
REGULAR INTERVALS BETWEEN OCCURRENCES

Edgar de Graaf   Walter A. Kosters

Leiden Institute of Advanced Computer Science
Leiden University, The Netherlands
{edegraaf,kosters}@liacs.nl

Abstract

We propose a new measure of support (the number of occurrences of a pattern in a dataset), where we count the number of times a pattern occurs (nearly) in the middle between two other occurrences. The more a pattern occurs in the middle of two occurrences (transaction wise) the more stable or regular the interval between occurrences is. We will define this measure and show how it can be implemented in ECLAT in order to mine for frequent patterns. Our goal is to find all patterns that occur with some fixed interval, e.g., a certain phone-call occurs almost every Friday.

1 Introduction

Mining frequent patterns is an important area of data mining where we discover substructures that occur often in (semi-)structured data. In this work we will further investigate one of the simplest structures: itemsets. Much research has been done in the area of frequent itemset mining. We will propose an algorithm that discovers patterns that occur at regular moments, or rather in regular intervals. This will enable us to mine for events that occur, e.g., every Friday. The technique is extendable to more complicated structures like sequences.

Stable patterns are patterns that occur frequently and with certain intervals. The *interval* is the number of transactions *without* the pattern in between two transactions (events) *with* the pattern. Instead of discovering stable patterns one could consider to add an extra item to the events, for example the day of the week. However by discovering stable patterns we hope to find more unexpected intervals, e.g., every three hours a pattern occurs. This type of pattern is interesting because it shows events occurring with a regular interval, e.g., every morning around 9:10 coffee and a cookie are bought.

We will define this type of support and show its usefulness. To this end, this paper makes the following contributions, with emphasis on the first and second:
— We will **define stable patterns and show that they possess the** APRIORI **property**. This means that if pattern p' is contained in pattern p, then the stability value of p is at most equal to that of p'. The property guarantees efficient mining implementations.
— Furthermore we will **propose an algorithm** for the discovery of stable patterns and discuss its efficiency.
— Finally we will show that **this enables us to find new and interesting patterns** via explorative experiments on real and synthetic datasets.

Our working example is the mining of an access log from the Computer Science department of Leiden University. This access log will first be converted to sets of properties we are interested in, e.g., pages visited every hour. From here on we call this dataset the *website* dataset.

This research is related to work done on the (re)definition of support, using time with patterns and the incorporation of distance measured by the number of transactions between pattern occurrences. The notion of support was first introduced by Agrawal et al. in [1] in 1993. Since then many new and faster algorithms where proposed. We make use of ECLAT, developed by Zaki et al. in [10]. Steinbach et al. in [8] generalized the notion of support providing a framework for different definitions of support in the future. Our work is also related to work described in [7] where association rules are mined that only occur within a certain time interval. Furthermore there is some minor relation with mining datastreams as described in [2, 6, 9], in the sense that they use time to say something about the importance of a pattern.

Finally this work is related to some of our earlier work. Results from [5] indicated that the biological problem could profit from incorporating consecutiveness into frequent itemset mining, which was elaborated in [3]. In the case of stable patterns we also make use of the transactions and the distance between them. Secondly in [4] it was mentioned that support is just another measure of saying how good a pattern fits with the data. There we defined different variations of this measure, and stability can been seen as one such variation.

The formal definitions concerning stable patterns and an algorithm are given in Section 2. In Section 3 we present experimental results, and we conclude in Section 4.

2 Stable Patterns

In this section we will define stable patterns. In particular, patterns that occur at regular intervals (e.g., at equidistant time stamps) will be called stable. In order to judge this property, we will determine how often events occur "in the middle" between two other events. We also describe an algorithm to find these patterns.

The intuitive idea of computing the variation in average distance between occurrences will not easily guarantee the APRIORI property, which is what we really need in the algorithm.

2.1 Definition

In this paper a dataset consists of a series of transactions that take zero time. Each transaction is an itemset, i.e., a subset of $\{1, 2, 3, \ldots, max\}$ for some fixed integer max. The transactions can have time stamps; if so, we assume that the transactions take place at different moments. We choose some notion of *distance* between transactions; examples include: (1) the distance is the time between the two transactions and (2) the distance is the number of transactions (in the *original* dataset) strictly in between the two transactions. We will define $\mathit{Trans}(p)$ as the series of transactions that contain pattern (i.e., itemset) p; the *support* of a pattern p is the number of elements in this ordered series.

We now define *w-stable patterns* as itemsets that occur frequent (support $\geq$ *minsup*) in the dataset and that have *stability value* $\geq$ *minstable*, where the values *minsup* and *minstable* are user defined thresholds. In order to define the stability value, we first introduce w-good triples: a *w-good triple* (L, M, R) consists of three transactions L, M and R, occurring in this order, such that $|\mathit{distance}(L, M) - \mathit{distance}(M, R)| \leq 2 \cdot w$; here w is a pregiven small constant ≥ 0, e.g., $w = 0$. The stability value of a pattern p is the number of w-good triples in $\mathit{Trans}(p)$, plus the number of transactions in $\mathit{Trans}(p)$ that occur as left endpoint in a w-good triple (the L mentioned above), plus the number of transactions in $\mathit{Trans}(p)$ that occur as right endpoint in a w-good triple (the R mentioned above).

Note that the stability value of a pattern p' with $p' \subseteq p$ is at least equal to that of p: the so-called APRIORI or anti-monotone property. Also note that the stability value remains the same if we consider the dataset in reverse order.

We now show that equidistant events are "very" stable (in case $w = 0$):

Theorem Suppose that $\mathit{Trans}(p)$ has n elements, so p has support n. If $\mathit{Trans}(p)$ satisfies:

1. $n - 2$ elements occur as the left endpoint of a 0-good triple,
2. $n - 2$ elements occur as the right endpoint of a 0-good triple, and
3. the number of 0-good triples equals $\lfloor n/2 \rfloor (\lceil n/2 \rceil - 1)$
   i.e., for even n: $n/2\,(n/2 - 1)$; for odd n: $((n-1)/2)^2$

then the transactions in $\mathit{Trans}(p)$ are *equidistant*. The values in 1, 2 and 3 are maximal, as is their sum.

Proof We proceed from the right (formally by induction). The end of the sequence $\mathit{Trans}(p) = (T_1, T_2, \ldots, T_n)$ looks like:

$\cdots$	$\mathcal{L}\&\mathcal{R}$	$\mathcal{L}\&\mathcal{R}$	$\mathcal{L}\&\mathcal{R}$	$\mathcal{L}\&\mathcal{R}$	$\mathcal{L}\&\mathcal{R}$	$\mathcal{R}$	$\mathcal{R}$
$\cdots$	T_{n-6}	T_{n-5}	T_{n-4}	T_{n-3}	T_{n-2}	T_{n-1}	T_n
$\cdots$	6	5	4	3	2	1	0

Here $\mathcal{L}/\mathcal{R}$ denotes: this T_i is a left/right endpoint in a 0-good triple; the numbers beneath the T_i's indicate the number of times T_i is the middle of a 0-good triple.

First observe T_{n-2}, T_{n-1} and T_n, where T_{n-2} is a left endpoint of a 0-good triple; this implies that $\textit{distance}(T_{n-2}, T_{n-1}) = \textit{distance}(T_{n-1}, T_n) = a$ for some a.

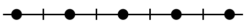
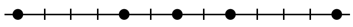
Now suppose we have the following situation: $T_i = L$ (with $i \geq \lfloor n/2 \rfloor$) is the left endpoint of a 0-good triple (L, M, R), for some $M = T_j$ with $j > i$; furthermore $a = \textit{distance}(T_\ell, T_{\ell+1})$ for all $\ell > i$. Now T_j occurs $n - j$ times as middle of 0-good triples, whose right endpoints are the consecutive $T_{j+1},\ldots,T_n$. We can conclude that $\textit{distance}(T_i, T_{i+1}) = a$. So we have $\textit{distance}(T_\ell, T_{\ell+1}) = a$ for $\ell = \lfloor n/2 \rfloor, \lfloor n/2 \rfloor + 1, \ldots, n$.

Similarly, using the right endpoints, one can show that $\textit{distance}(T_\ell, T_{\ell+1}) = b$ for some b ($\ell = 1, 2, \ldots, \lceil n/2 \rceil$). Using $\ell = \lfloor n/2 \rfloor$ we see that $a = b$. □

Example 1 Suppose we have the following 9 itemsets in our dataset:

transaction 1: $\{A,\ B,\ C\}$
transaction 2: $\{D,\ C\}$
transaction 3: $\{A,\ B,\ E\}$
transaction 4: $\{E,\ F\}$
transaction 5: $\{A,\ B,\ F\}$
transaction 6: $\{E,\ F\}$
transaction 7: $\{A,\ B,\ F\}$
transaction 8: $\{E,\ F\}$
transaction 9: $\{A,\ B,\ C\}$

As distance we take the number of intermediate transactions. The stability value (with $w = 0$) of $\{A,B\}$ is $4 + 3 + 3 = 10$, the maximal value possible: there are 4 0-good triples; we have 3 transactions that are left (right) endpoint of a 0-good triple (see picture below, left; dots indicate the transactions that include $\{A,B\}$). If we insert two transactions $\{E,\ F\}$ between transaction 1 and 2, and also two between 8 and 9, we still have 4 0-good triples, but now we only have 2 transactions that are left (right) endpoint of a good 0-triple (see picture below, right), leading to stability value $4 + 2 + 2 = 8 < 10$. This example shows that condition 3 from the Theorem is in itself not sufficient yet in order to guarantee equidistance.

2.2 Algorithm

We now consider algorithms that find all stable patterns, given a dataset. Thanks to the Apriori property many efficient algorithms do exist. However, the really fast ones rely upon the concept of FP-tree or something similar, which does not keep track of stability. This makes these algorithms hard to adapt for discovering stable patterns.

Eclat [10] is a fast algorithm that does not make use of FP-trees; it grows patterns recursively while remembering which transactions contained the pattern, making it very suitable for our purpose. In a recursive step only these transactions are considered when counting the occurrence of a pattern. All counting is done by using a matrix and patterns are extended with new items using the order in the matrix. This can easily be adapted to incorporate stability.

Now suppose that $\textit{Trans}(p) = T^{parent}$, with $n = |T^{parent}|$, is the ordered series of transactions (augmented with their index numbers from the *original* dataset) that contain itemset p. The algorithm below (Algorithm 1) will calculate the stability value when adding a new *item* to p. The algorithm will also calculate the support and the new series of transactions T^{child} that will be considered in the next step of a frequent pattern mining algorithm: the Eclat algorithm is extended to StableClat. The *child* is the *parent* itemset p extended with the new *item*. Note that *Left* and *Right* are *sets*. In line (9) we add the *index numbers* of the transactions. The function $\textit{contains}(\textit{trans}, \textit{item})$ checks if the transaction *trans* contains the item *item*, the function $\textit{has}(T^{parent}, \textit{index})$ verifies that transaction *index* is in T^{parent}; $T_{\textit{index}}$ is the transaction as retrieved from the original dataset. The *mindepth* threshold defines from which depth the stability should be calculated — otherwise, for small itemsets with large supports the computation would become cumbersome. The *depth* is the recursive depth that is equal to the size of the child pattern that we are considering.

Algorithm 1 Stability Value

```
$support := 0$, $stable := 0$, $Left := \emptyset$, $Right := \emptyset$, $T^{child}$ := empty series, $i := 1$
while $i \leq n$ do
    if $contains(T_i^{parent}, item)$ then
        $T^{child} := T^{child}$ with $T_i^{parent}$ appended, $support := support + 1$
        if $depth \geq mindepth$ then
            $j := i + 2$
            while $j \leq n$ do
                if $contains(T_j^{parent}, item)$ then
                    $middle := (T_i^{parent} + T_j^{parent})$ mod 2, $index := (T_i^{parent} + T_j^{parent}) \,/\, 2$
                    if $middle = 0$ and $has(T^{parent}, index)$ and $contains(T_{index}, item)$ then
                        $stable := stable + 1$
                        $Left := Left \cup \{T_i^{parent}\}$, $Right := Right \cup \{T_j^{parent}\}$
                    end if
                end if
                $j := j + 1$
            end while
        end if
    end if
    $i := i + 1$
end while
$stable := stable + |Left| + |Right|$
```

This algorithm will only increase *stable* if the pattern is *exactly* in the center of two transactions containing the pattern (so $w = 0$); this can be easily generalized. It is possible that the pattern doesn't occur in the center transaction but in a transaction that is very near. A threshold $w > 0$ can be specified, as in the previous section. Suppose T_i is the outer left transaction and T_j is the outer right transaction, then we consider every T_ℓ, where $i < (i+j)/2 - w \leq \ell \leq (i+j)/2 + w < j$. Now our algorithm needs to check if the pattern occurs in one of these transactions.

Example 2 Suppose we have the same 9 transactions as in the previous example. If $w = 1$ then the stability value of $\{A,B\}$ will be $10 + 2 = 12$: transactions 1 and 7 (and 3 and 9) are now also endpoints of a 1-good triple.

The maximal stability value depends on the size of the database. This makes setting the minimal stability threshold *minstable* somewhat difficult. To make it easier one only needs to give a number *dist*, where $dist > 0$. With *dist* we calculate the *stable* value if the distance between all transactions containing the pattern is precisely *dist*. In this calculation we disregard the count of the left and right endpoints. This *dist* can now be used to propose a reasonable value for *minstable*, where D is the original dataset:

$$minstable = \binom{\lceil |D|/dist \rceil}{2}$$

Most frequent itemsets or patterns have a high stability value because it is more likely that a center transaction contains the pattern. However these patterns will not necessarily make a stable pattern more apparent. Furthermore it might also be contained in many transactions that don't form a stable interval. In order to solve this problem we can divide the stability value by the square of *support* and let $newstable = stable/support^2$. However we will lose the anti-monotone property, so this will only be useful as a post-processing step. We choose to divide by $support^2$ because stable can maximally become

$$\binom{support}{2} + 2 \cdot (support - 2) < support^2$$

In such a way we remove the influence of a high support on stability.

3 Results and Performance

The experiments were done for three main reasons, and are of exploratory nature. First of all by using the synthetic dataset we *show that patterns with a stable interval will be found.* Secondly with the website dataset we *show that the algorithm also finds good stable patterns for real problems.* And finally with a synthetic dataset and with the website dataset we *want to examine the efficiency of the algorithm compared to normal* Eclat. Of course the normal Eclat algorithm only finds frequent itemsets and no stable patterns. However the goal is to show the influence of the search for stable patterns on speed. Our implementation of Eclat that discovers stable patterns is called StableClat. All experiments were done on a Pentium 4 2.8 GHz with 512MB RAM.

The synthetic datasets can be seen as a supermarket that sells newspapers and credits for cell phones. The combination of the two is sold every day in the morning at least x times. The first dataset contains 1,000 transactions and 110 items. Of these 110 items 10 occur every 4 transactions. Also each item has a support of 200. From here on we call this dataset *news&credit small.* The second dataset contains 5,000 transactions and 110 items. Of these 110 items 10 occur every 10 transactions. Also each item has a support of 1,000. From here on we call this dataset *news&credit large.* The StableClat algorithm was also tested on a real dataset. This dataset is based on an access log of the website of the Computer Science department of Leiden University, as said before. It contains all 1,991 items of the webpages that were visited, grouped in one hour blocks, so each of the 744 transactions contains the pages visited during one hour. This dataset will be called *website.*

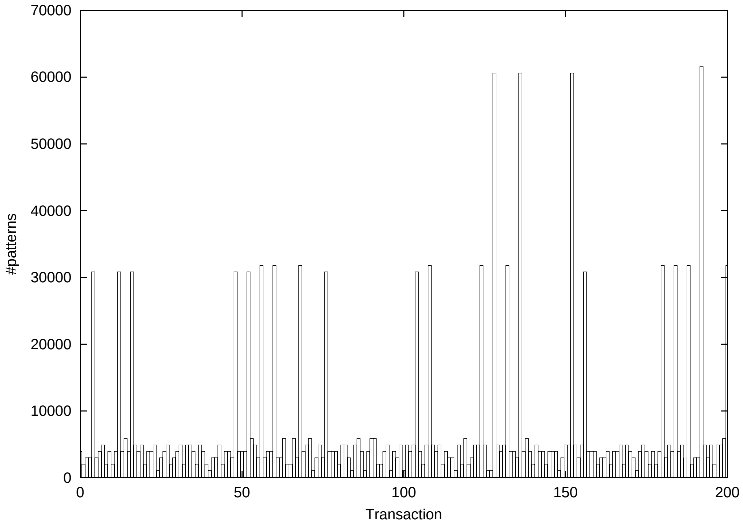


Figure 1: Occurrence graph of news&credit small using traditional support ($minsup = 25$)

Figure 1 and Figure 2 show an *occurrence graph.* For each transaction the number of frequent patterns contained by it are counted and plotted. The use of a minimal stability threshold will give less patterns because it filters out non-stable patterns. The items in the news&credit small dataset all occur 200 times, so their support is 200. Usually transactions are made up out of 20 randomly selected items. However every 4 transactions we randomly select 10 items and the remaining 10 items are the items that will be in the stable pattern. In this way there are several stable patterns, but also many unstable ones. Figure 1 shows that with only support we will not discover these stable patterns easily. When we use a minimal stability threshold (with the *dist* parameter from Section 2) we are able to see these patterns as we show with Figure 2. In all experiments we let $mindepth = 2$.

Figure 3 shows the occurrence of one pattern in the first 100 transactions from the website dataset. Transaction 0 is on Saturday 1 May 2004 5:00 and transaction 100 is on Wednesday 5 May 2004 9:00. First we searched the patterns with minimal stability value $\binom{\lceil 744/7 \rceil}{2}$. Then we selected one pattern where the fraction $stable/support^2$ is maximal. The figure shows a regularity in the occurrence of the selected pattern, which consists of four personal homepages and a main page.

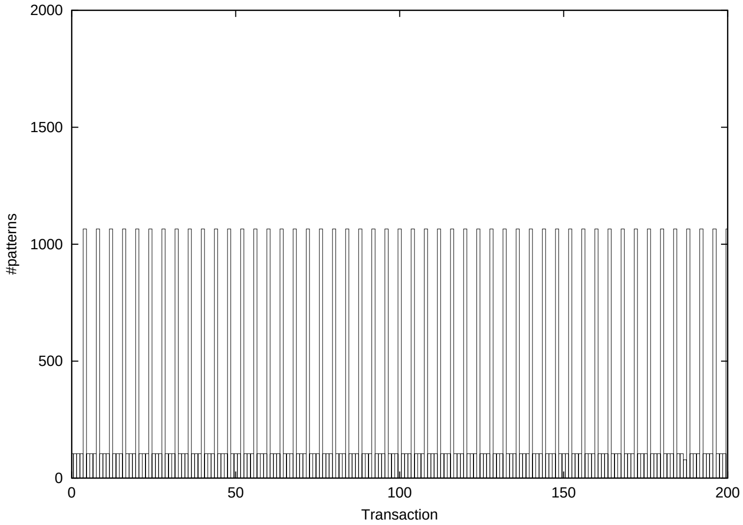


Figure 2: Occurrence graph of news&credit small using traditional support *and* stability ($minsup = 25$, $dist = 20$)

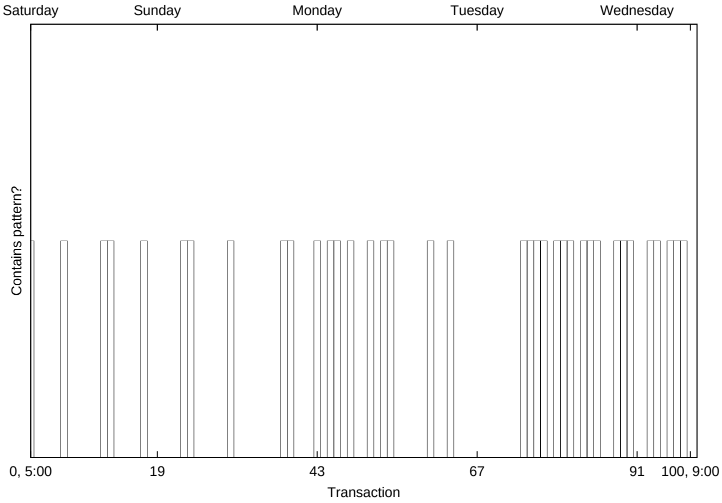


Figure 3: The occurrence of one stable pattern for website ($minsup = 100$, $dist = 7$)

	news&credit small	news&credit large	website
support only	1	36	179
with stability	2	140	251

Table 1: Time in seconds needed to mine each of the three datasets ($minsup = 100$, $dist = 4$)

Table 1 gives an indication of the influence of stability calculation on the speed of the algorithm. The news&credit large dataset shows a clear slowdown because of the occurrence of more frequent patterns. This means that more combinations have to be checked for a pattern occurring in the center transaction of a series of transactions, which requires an amount of time that is quadratic in the number of elements in this series.

4 Conclusions and Future Work

When we use stability in our search for patterns, we are able find patterns that occur with some regular interval. The measure we proposed in this paper still enables us to prune using anti-monotonicity.

Using the distance between transactions like it is done in this paper is an interesting area of research. In the future we want to examine new measures that would enable us to visualize other types of behavior. Also we want to see if we can speed up the search for stable patterns, e.g., by using heuristics or by improving the stability measure. Furthermore, we would like to compare our approach with post-processing methods.

Acknowledgments This research is carried out within the Netherlands Organization for Scientific Research (NWO) MISTA Project (grant no. 612.066.304). We would like to thank Carlos Soares.

References

[1] Agrawal, R., Imielinski, T., Srikant, R.: *Mining Association Rules between Sets of Items in Large Databases.* In Proc. ACM SIGMOD Conference on Management of Data (1993), pp. 207–216.

[2] Chen, Y., Dong G., Han J., Wah, B., Wang J.: *Multidimensional Regression Analysis of Time-series Data Streams.* In Proc. 28th Int. Conference on Very Large Data Bases (VLDB 2002), pp. 323–334.

[3] Graaf, E.H. de, Graaf, J.M. de, Kosters, W.A.: *Using Consecutive Support for Genomic Profiling.* ECML/PKDD-2006 Workshop on Data and Text Mining for Integrative Biology, Berlin, Germany, 2006.

[4] Graaf, E.H. de, Kosters, W.A.: *Efficient Feature Detection for Sequence Classification in a Receptor Database.* In Proc. Seventeenth Belgium-Netherlands Conference on Artificial Intelligence (BNAIC 2005), pp. 81–88.

[5] Graaf, J.M. de, Menezes, R.X. de, Boer, J.M., Kosters, W.A.: *Frequent Itemsets for Genomic Profiling.* In Proc. 1st International Symposium on Computational Life Sciences (CompLife 2005), LNCS 3695, Springer, pp. 104–116.

[6] Giannella, C., Han, J., Pei J., Yan, X., Yu, P.: *Mining Frequent Patterns in Data Streams at Multiple Time Granularities.* In Proceedings of the NSF Workshop on Next Generation Data Mining (NGDM 2002), pp.191–210.

[7] Li, Y., Ning, P., Wang, X.S., Jajodia, S.: *Discovering Calendar-based Temporal Association Rules.* In Proc. of the 8th Int Symposium on Temporal Representation and Reasoning (TIME 2001), pp. 111–118.

[8] Steinbach, M., Tan, P., Xiong, H., Kumar, V.: *Generalizing the Notion of Support.* In Proc. 10th Int. Conf. on Knowledge Discovery and Data Mining (KDD 2004), pp. 689–694.

[9] Teng, W., Chen, M., Yu, P.S.: *A Regression-based Temporal Pattern Mining Scheme for Data Streams.* In Proc. 29th Int. Conference on Very Large Data Bases (VLDB 2003), pp. 93–104.

[10] Zaki, M., Parthasarathy, S., Ogihara, M., Li, W.: *New Algorithms for Fast Discovery of Association Rules.* In Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining (KDD 1997), pp. 283–296.