Sequential Pattern Mining for Uncertain Data Streams using Sequential Sketch

Jingyu Chen  
School of Computer Science and Technology, Xidian University, Xi’an, China  
Email: jychen@mail.xidian.edu.cn

Ping Chen  
Software Engineering Institute, Xidian University, Xi’an, China  
Email: chenping@mail.xidian.edu.cn

Abstract—Uncertainty is inherent in data streams, and present new challenges to data streams mining. For continuous arriving and large size of data streams, modeling sequences of uncertain time series data streams require significantly more space. Therefore, it is important to construct compressed representation for storing uncertain time series data. Based on granules, sequential sketches are created to store hash-compressed granules. And based on sliding windows, a sketch update strategy is given to store most recent granules. As the sequential sketches may be saturated with the increasing of uncertain data streams, this paper presents an optimization strategy to delete the absolute sparse patterns. Based on the sequential sketches, a sequential pattern mining algorithm is proposed for mining uncertain data streams. The experimental results illustrate the effectiveness of the pattern mining algorithm.

Index Terms—Sketch; Sequential Pattern; Granulation; Uncertain Time Series; Data Stream

I. INTRODUCTION

With the developing of information technique and wireless sensor networks, a great variety application need to analysis and mining large-scale, continuous and rapid arriving data streams. As a common data form in data streams, time series data streams contain much more useful pattern and information for many applications fields.

Time series is a series of observation data according to a certain time sequence, and aggregates with time and event [2]. Time-series data mining is an important way which mining some useful and potential knowledge from a great deal of time-series. Time series data are created by many applications such as RFID, traffic, mobile, economic and finance applications. In these applications, uncertainty often happens because of network failure, noise and sampling error, etc. Probability time series are common uncertain time series data. Since there are many possible observation values at a time in the uncertain time series, it may generate much more possible combination of sequences and make sequences model much complex. The probabilistic information should be handled in sequential pattern mining uncertain time series data streams.

As an important issue in data mining, frequent pattern mining has been widely applied in the traditional databases. However, the uncertainty in many data streams applications has brought great challenges for frequent pattern mining. In many practical applications, the data are often collected and stored in the form of probability, thus increase the possible data patterns, and bring non-ignorable effect on the results of pattern mining. Specifically for the uncertain data streams, the mining algorithm should handle the continuous arriving streams real-time; it presents a huge challenge for frequent pattern mining in uncertain data streams. Therefore, designing efficient compressed storage and mining sequential patterns of the probability data streams become the research hotspots of frequent pattern mining.

To address the problem of uncertain sequential patterns mining we suggest a hash based sequential sketches approach to reduce storage spaces and computational complexity. The approach is based on granulation mechanism, which granulates uncertain time series data to a set of possible sequences. And we use sketch technique to compress all possible sequences to sequential sketches. As the sequential sketches may be saturated with the increasing of uncertain data streams, an optimization strategy is designed to delete the absolute sparse patterns. Based on the sequential sketches, a sequential pattern mining algorithm is proposed for mining uncertain time series data streams. Final, we verify the accuracy and efficiency of the proposed scheme via experiments.

The rest of the paper is organized as follows. Section II discusses related work. Section III presents a sequential sketch model for uncertain time series data. Section IV outlines the sequential sketches based pattern mining methods. Simulation methodology and performance evaluation result and analysis are presented in section V, and we conclude the work in section VI.

II. RELATED WORK

Time series data mining has been attracting much attention in research and practice. As a hot research area in time series data mining, frequent pattern mining can be mainly classified into two broad categories: Apriori-
based [5-7] and Tree-based [8-10] algorithms. As an inherent property of data streams, uncertainty brings great challenge to data streams mining.

Uncertain time series often have same characteristics as uncertain data streams, such as uncertain, large-scale, continuous and rapid arriving. To handle the continuous arriving uncertain time series streams, we often expand some streams mining algorithms to mine frequent patterns uncertain streams. In recent years there have been a plethora of methods for managing and mining uncertain data streams. Several important stream pattern mining algorithms have been introduced in recent years. Chui et al. present an uncertain data model and propose a U-Apriori to mine frequent itemsets from uncertain data [11]. Based on FP-growth [8], Leung et al. propose two tree-based mining algorithms UF-Streaming and SUF-growth to efficiently find frequent itemsets from streams of uncertain data, where each item in the transactions in the streams is associated with an existential probability [1]. Kaneiwa et al. propose a method for mining such local patterns from sequences by using rough set theory [12]. Ackermann et al. present a new corsets trees based clustering algorithm to improve quality of stream clustering [13]. Tran et al. present the PODS model for processing uncertain data using continuous random variables [14]. Nie et al. employ a time-varying graph model to represent imprecise object relationships with compression, and present a probabilistic algorithm to estimate the most likely location [15]. Lian et al. formalize and guarantee the accuracy of join on uncertain data streams, and propose effective pruning methods to filter out false alarms [16].

Sketch is a popular method for handling huge and fast data streams. Sketch techniques use a sketch vector as a data structure to store the streaming data compactly in a small-memory footprint. The main advantage of using these sketch techniques [17, 18] is that they require a storage which is significantly smaller than the input stream length. Sketch techniques are used in stream data frequent items mining [19, 20], clustering [21] and anomaly detection [22, 23] recently.

Our work is closely related to mine frequent patterns of uncertain time series data. In this paper, we model the uncertain time series data based on sequential sketch. We use a sequential sketch approach to create hash-compressed representations. We also design an optimization strategy to avoid the sequential sketch saturated. Then, we propose a sequential pattern mining algorithm to mine frequent sequential patterns of the uncertain time series data streams.

III. SEQUENTIAL SKETCH

One of the most effective ways to deal with imprecise and uncertain data is to employ probabilistic approaches. Since we may get several probabilistic points at a certain time t, the probability time-series data may have much more size of data. The probability also increases the complexity of modeling and analysis of time series data. The way of modeling the probability of data is a key point of uncertain data stream mining.

Probability time series is a kind of uncertain time series data, which has time, event and event’s probability. We use the following definition to represent and store the probability time series data.

Definition 1. (Element) An element e = <x,p,t> is a basic element of probability time series data, where x represents an observation value, p represent the probability of x value, and t represents observation time.

Definition 2. (Item) An item I is a sequence s at time t includes all possible elements at time t. I = {<s1,p1,t>, <s2,p2,t>, ……}

Definition 3. (n-length uncertain time series) A uncertain time series utn is an ordered list of items that include n consecutive items of uncertain time series, utn={I1, I2, ……In}.

Definition 4. (Probability of sequence) A probability ps(s, utn) of a sequence s in a uncertain time series ut is the probabilities multiplication of each elements of s in utn.

\[ ps(s, utn) = \prod_{e \in s} p(e) \] where \( s_i \subseteq ut_n \), II means multiplication, e represents a element in sequence s, and \( p() \) represents the probability value of element.

Examples of uncertain time series data are showed in Table1.

| Table1. Examples of Uncertain Time Series Data |
|--------|--------|--------|--------|--------|--------|
| x  | p  | t  | x  | p  | t  |
| a  | 0.6 | 1  | c  | 0.2 | 4  | f  | 0.4 | 7  |
| b  | 0.4 | 1  | d  | 0.4 | 4  | e  | 0.6 | 7  |
| a  | 0.5 | 2  | e  | 0.4 | 4  | g  | 0.2 | 8  |
| b  | 0.5 | 2  | d  | 0.8 | 5  | b  | 0.5 | 8  |
| a  | 0.3 | 3  | c  | 0.2 | 5  | d  | 0.3 | 8  |
| b  | 0.4 | 3  | c  | 0.5 | 6  | d  | 0.7 | 9  |
| e  | 0.3 | 3  | d  | 0.5 | 6  | f  | 0.3 | 9  |

In Table1, there are two possible elements \( I_i=\{a,0.6,1>, b,0.4,1> \) at the time t=1.

The uncertain time series of \( ut_i \) is

\[ I_1(I_2, I_3) = \{ \begin{array}{lll} a : 0.6 & a : 0.5 & a : 0.3 \\ b : 0.4 & b : 0.5 & b : 0.4 \\ e : 0.3 \end{array} \} \]

In Table 1 at time t=1, \( I_i \) has two possible values a and b, and the possibilities values of a and b are \( p(a)=0.6 \) and \( p(b)=0.4 \). For \( ut_i \) from time t=1 to t=2, \( ps(ab, ut_3) = 0.6^*0.4 = 0.24 \).

The probability increases difficulty for analysis uncertain time series, for it generates much more possible combination of sequences and make sequences model much complex.

A. Granularities of Uncertain Sequences

Granular cognition plays an important role for complex data modeling. The mechanism of granulation has been applied in many areas of reasoning under uncertainty [24]. The granules may simplify and speed up the computational tasks such as: searching, mining, or reasoning. As the basic element for analyzing and mining time series data, sequences and subsequences of time series also are basic element for granularity construction.
We use the granular mechanism to model and represent the complex uncertain time series data.

To deal with the diversity of uncertain sequential data, we describe sequences of time points using a set of granules. We consider the different length of subsequences time points as different granules. We define the following concepts for uncertain time series granulating.

Definition 5. \((m\text{-length subsequences set})\) A \(m\)-length subsequences set \(us_m\) of an uncertain time series \(ut_n\) is a set of all possible \(m\)-length subsequences of the uncertain time series.

\[
us_m = \{ s | s \subseteq ut_n \& \|s\| = m \}
\]

where \(\subseteq\) means \(s\) is a subsequence of \(ut_n\) and \(\|\|\) represents the length of a sequence.

For example, for the uncertain time series \(ut_n\), at time 1 and 2, we can get a 2-length subsequences set \(us_2 = \{ aa, ab, ba, bb \}\) and the possibilities set of each sequence in \(us_2\) is \(\{0.3, 0.3, 0.2, 0.2\}\). At continuous time 1, 2 and 3, \(us_3 = \{aaa, aab, aae, aba, abb, aab, bba, bab, bae, bba, bbe\}\) and the possibilities set of each sequence in \(us_3\) is \(\{0.09, 0.12, 0.09, 0.12, 0.09, 0.06, 0.08, 0.06, 0.08, 0.06, 0.08, 0.06\}\).

For counting frequent patterns, the number of subsequences and the positions of subsequences in sequences are both important. So, we store both number and positions of subsequences in our sketches.

Definition 6. \((k\text{-Granule})\) A \(k\)-granule \(G_k\) \((\text{sub}, \text{ut}_n)\) includes a set of \(\text{positions}\) of the corresponding subsequence \(\text{sub}\) \(n\)-length uncertain time series \(\text{ut}_n\) and the possibility value of subsequences \(\text{sub}\) should not smaller than \(\gamma=0.2\). The \(\gamma\) can be defined by users.

For example, in table 1, the 2-Granule \(G_2\) \((\text{ab}, \text{ut}_1)\) of uncertain time series \(\text{ut}_1\) is the positions set \(\{1, 2\}\) and the probability value of subsequence \(\text{ab}\) is \(\{0.3, 0.2\}\) (all greater than \(\gamma=0.2\)).

Based on the \(k\)-granule, we can group subsequences of uncertain time series streams into different set of granules. For the diversity of \(k\)-Granule, we also need to compress the storage space. We construct sketches to storage \(k\)-Granules compressed, and calculate approximately the similarity between objects’ time series by testing similarity between these sketches.

B. Sequential Sketch

Sketch based approaches [14] were designed for enumeration of different kinds of frequency statistics of data sets. A commonly-used sketch is the count-min method [14]. The count-min sketch use \(w = \lfloor \ln(1/\delta) \rfloor\) pairwise independent hash functions, each hash function maps data into uniformly random integer in the range \(h = [0, e/c]\), where \(e\) is the base of the natural logarithm. The data structure itself consists of a two dimensional array with \(w\)-h cells with a length of \(h\) and width of \(w\). Each hash function corresponds to one of \(w\) 1-dimensional arrays with \(h\) cells each. In standard applications of the count-min sketch, the hash functions are used in order to update the counts of the different cells in this 2-dimensional data structure.

Definition 7. \((\text{Sequential sketch})\) A sequential sketch includes a two-dimensional matrix \(SK[w, e/e/c] (w<<N, c)\), \(c = e/e/c\) is the maximum value of hash value range and \(w\) is the number of hash functions \(hr[w]\). The sequential patterns of the \(k\)-granules set are denoted by \(SP\), such as \(\{aa, ab, \ldots\}\). Let \(hr\) be the \(i\)-th hash function in \(hr[w]\): \(SP \rightarrow \{0, \ldots, c\}\) be a hash function that hash a sequential pattern and its possibility value to the \(i\)-th row number and store the possibility value at the \(hr\) column. The initial value of each element in the sequential sketch is 0. For each sequential pattern \(sp\) in the \(k\)-granules set, we add possibility value \(psl\) of \(sp\) in the granules to \(SK[i, hr[i](sp)]\).

\[
SK[i, hr(sp)] = SK[i, hr(sp)] + pl(sp)
\]

Figure 1 shows the update process of the sequential sketch.

We adopt sliding windows technique to process the continuous arriving time series data streams. Assuming the size of each sliding window is \(L\) and the maximum length of sequences of all time series is \(n\) in a sliding window. In a time window \(w_t\), we construct \(n\)-sequential sketch for a time series \(s_t\) to reduce the scale of uncertain time series data. We use a set of \(n\)-sequential sketches \(SK_s = \{SK_0, SK_1, \ldots, SK_{n-1}\}\). We show an example of \(m\) sequential sketches in figure 2.

For granules in an uncertain time series, if any sequence has \(j\) position, the sequence and its possibility will be store in the \(j\)-th sequential sketch \(SK_j\).

For example, in table 1, the 2-Granule \(G_2\) \((\text{ab}, \text{ut}_1)\) of uncertain time series \(\text{ut}_1\) is the positions set \(\{1, 2\}\). So \(ab\) will be store at \(SK_1\) and \(SK_2\).

We compress all possible sequences of uncertain time series to \(n\)-sequential sketches according the positions of all \(k\)-granules of a possible sequence.

Rather than store and count all possible sequences, we calculate the possible sequential patterns on the sequential sketches.

In the \(n\)-sequential sketches, each sequential sketch \(SK_i\) represents all sequential patterns that appear at \(j\)-th position of uncertain time series data. Using sequential sketches, we can calculate the support and sketch support of a certain position for each pattern. The sketch support value \(sk_{\text{sup}}\) for a pattern \(sp\) at a position \(j\) is the minimum value of the \(h_i(sp)\) at \(SK_j\), where \(i\) is the \(i\)-th hash function \((0 \leq i \leq w-1)\). The sketch support value \(sk_{\text{sup}}\) of a pattern \(sp\) at position \(j\) is defined by
\( sk\_sup(sp, j) = \min(SK_j[1, h_0(sp)], ..., SK_j[i, h_{n}\_sp(sp)]) \) (3)

where \( \min \) is a function for minimum value.

The support value \( Sup \) for a pattern \( sp \) is the sum of \( sk\_sup \) at each sequential sketch.

\[
Sup(sp) = \sum_{j=0}^{n-1} sk\_sup(sp, j)
\]

(4)

According to the characteristics of the uncertain data stream, we define uncertain frequent pattern for time series pattern mining.

Definition 8. (Frequent pattern). A pattern \( sp \) is an frequent pattern, if its support value \( Sup(sp) \) is larger than \( minSup \), where \( minSup \) \((0 < minSup < 1) \) is a user-defined parameter.

Definition 9. (Sparse pattern). A pattern \( sp \) is a sparse pattern, if \( Sup(sp)/n \) is smaller than \( minSup \), where \( n \) is the number of sequential sketches.

To store the pattern information, we also construct a patterns array \( GA_j[n] \) to store frequent patterns for each sequential sketch \( SK_j \), where \( n \) is the maximum length of sequences of all time series in a sliding window. We use a set of \( n-1 \) patterns array \( GAS= \{GA_0, GA_1, ..., GA_{n-1}\} \), and each \( GA_j \) store the frequent patterns at \( j \)-th position of sequences for each sequential sketch \( SK_j \).

The initial value of each element in granular array \( GA_j[n] \) is null. We construct the granular array \( GA_j \) for each sketch in then following strategy.

When a \( k \)-granule \( g_i \) is added to sequential sketches, for the pattern \( sp \) in \( g_i \), we check the corresponding sequential sketches and count the sketches supports \( sk\_sup \) of \( sp \) and the support \( Sup(sp) \).

For a sequential sketch \( SK_j \), if \( GA_j \) is not full and if \( sk\_sup(sp, j) > minSup \), we add \( sp \) and its sketch support value \( sk\_sup(sp, j) \) in \( GA_j \);

If \( GA_j \) is full, we compare the \( sk\_sup(sp, j) \) with the minimal \( sk\_sup(sj, j) \), where \( sj \) is a pattern in \( GA_j \).

If \( sk\_sup(sp, j) > sk\_sup(sj, j) \), then replace \( sj \) with \( sp \) and \( sk\_sup(sp, j) \).

The pattern array \( GA_j \) will store the top-\( n \) frequent patterns of \( k \)-granules in \( j \)-th position of an uncertain time series.

For the probability of uncertain time series, sequences may have various combinations of elements, and may generate numerous possible sequences. Based on the pattern arrays, we can construct a tree of sequential patterns to increase the length of sequential patterns. The tree of sequential patterns is show in figure 3.

As the size of the sequential sketches is much less than the size of primitive uncertain data streams. As time series data arriving, the sketch core may be filled full and saturated. Therefore, we adopt optimization technique to delete the absolute sparse pattern in the sequential sketches.

IV. SEQUENTIAL PATTERN MINING ALGORITHM

With the uncertain data streams continued arrival, the number of sequential patterns stored in the sequential sketch and pattern arrays is increasing rapidly and makes the sketch saturated. Therefore, it is important to delete those sparse sequential patterns.

A. Absolute Sparse Pattern

Due to the fast-changing characteristics of the data streams, some sparse patterns that were deleted quickly appeared in large numbers, even grown into frequent patterns. Therefore, some sparse patterns which have potential growth should be retained. We analyze the potential growth of sparse patterns and divide them into four categories:

1. Some patterns appear in small number and always belong to the sparse mode;
2. Some patterns used to be non-sparse patterns. Later due to small number of apperence, those patterns degraded to sparse patterns, and keep to be sparse patterns for a long time;
3. Some patterns used to be non-sparse patterns and degraded to sparse patterns. Then, those patterns appear frequently and have potential growth;
4. Some patterns are new patterns, but appear frequently and have potential growth.

From the above four categories, we can see the potential growth of categories 1) and 2) are very small. We should recognize those patents that belong to categories 1) and 2) as absolute sparse patterns and delete it form the sequential sketch.

We identify absolute sparse patterns according the changes of patterns over time. For each pattern we calculate its shortest time interval \( \delta_0 \) that degraded to sparse patterns.

\[
\delta_0 = \frac{1}{\lambda} \log_\lambda \left( \frac{\min Sup - \varepsilon}{(\min Sup - \varepsilon) - 1} \right)
\]

(5)

where \( \varepsilon \) \((0 < \varepsilon < minSup) \) is an error factor and \( \lambda \) is a attenuation coefficient. The \( \varepsilon \) and \( \lambda \) can be defined according the following formulas.

\[
\varepsilon = \frac{1}{3} \min Sup
\]

(6)

\[
0 \leq \lambda \leq \log_\frac{1}{\varepsilon} \left( \frac{\min Sup}{\varepsilon} \right)^* \frac{2}{2T-1-\min Sup}
\]

(7)

For a new pattern, \( \delta_1 \) is the shortest time interval for the new pattern to become a non-sparse pattern.
For each patterns in pattern arrays GA, we add two columns to store the creation time \( t_c \) and the most recent modification time \( t_m \). Let \( T_F = d_0 + d_1 \) be the degraded period of a pattern. For each pattern \( sp \) in a GA, from its \( t_c \) to \( t_m \), if its support is \( \text{Sup}(sp) / n < \varepsilon \), then it will be a sparse pattern. For a sparse pattern \( sp \), if its support is still \( \text{Sup}(sp) / n < \varepsilon \) during future time period \([t_c, t_c + T_F]\), the sparse pattern \( sp \) will become an absolute sparse pattern. For an absolute sparse pattern, the all value of row \( h(sp) \) in the sequential sketch \( SK_i \) will be deleted, and the absolute sparse pattern will also be deleted from the pattern array \( GA_i \). And the corresponding nodes will also be pruned from the sequential patterns tree.

**B. Sequential Pattern Mining Algorithm**

We adopt sliding windows technique to process the continuous arriving time series data. The size of each sliding window is \( L \). Let \( B[i] \) be the \( i \)-th sliding windows, \( B[1] \) is the first window, and \( B[L] \) be the most recent window.

According the sequential sketches and optimization strategy constructed above, we proposed a new sequential pattern mining algorithm \( UG-Miner \) based on sliding windows for uncertain time series data streams.

The algorithm of \( UG-Miner \) is shown as following:

**Algorithm of sequential pattern mining algorithm**

**Input:**
- uncertain time data series stream \( at \)
- \( n \)-1 sequential sketches \( SK[w,c] \) and \( n \)-1 pattern array \( GA_i \) [\( w \)]
- sliding windows \( B[1], B[2]\)...
- user-given parameters: \( \gamma, \varepsilon \)
- user-given the minimal support threshold \( \text{minSup} \)

**Output:**
- A set of sequential patterns: \( fpset \)

for a new coming sliding window \( B[i] \)
- gain uncertain time series \( at_i \) from stream
- for each 2-granule \( g \), in \( at_i \)
- for all positions \( j \) of pattern \( sp \) of 2-granule \( g \), store \( sp \) at \( SK_g \)
- if \( \text{sk_sup}(sp,j) < \text{minSup} \) and \( sp \) is not in \( GA_i \)
  - add \( sp \), possibility value and time to the \( GA_i \); add \( sp \) to the sequential patterns tree;
  - if \( \text{Sup}(sp)/n < \varepsilon \)
    - mark \( sp \) as a sparse pattern in \( GA_i \);
    - next granule;
  - for each \( GA_i \)
    - if the time interval \( > T_F \)
      - delete \( sp \) as the absolute sparse pattern \( sp \);
    - delete \( sp \) in all \( SK \) and \( GA \);
    - delete \( sp \) in the sequential patterns tree;
    - next \( GA_i \)
    - if half of granules in \( at_i \) are not add to \( fpset \)
      - increase \( \text{minSup} = \text{minSup} + \text{minSup} \times \varepsilon \);
    - if all granules in \( at_i \) are add to \( fpset \)
      - decrease \( \text{minSup} = \text{minSup} - \text{minSup} \times \varepsilon \);
    - next sliding window;

The key steps of the \( UG-Miner \) algorithm are check and count the possibility value in the sequential sketch to approximately statistics sequential patterns. Then, the \( UG-Miner \) algorithm updates the sequential sketch and the pattern arrays according the number of sequential patterns and optimization strategy. Based on the patterns arrays, a sequential tree is constructed and updated to gain more and longer sequential patterns.

**V. EXPERIMENT**

This section shows the results from our experiment to validate the accuracy and efficiency of our proposed sequential pattern mining algorithm.

We will show the accuracy of our algorithm based on the evaluation of coverage. Cover is the coverage of the algorithm results,

\[
\text{Cover} = \frac{|C_M \cap C_I|}{C_I} \quad (9)
\]

where \( C_M \) is the results frequent patterns set of \( UG-Miner \) algorithm, and \( C_I \) is the real frequent patterns. We compare our \( UG-Miner \) algorithm to \( SUF-Growth \) algorithms by using the datasets generated by the program developed at IBM Almaden Research Center [5]. We assigned an probability following the normal distribution from the range \((0.1)\) to each item in the data set.

Figure 4 shows the comparison of coverage between \( UG-Miner \) and \( SUF-Growth \) with increasing size of data set. In figure 4, the coverage of \( SUF-Growth \) is higher than that of \( UG-Miner \), because the \( UG-Miner \) is an approximate algorithm and use hash compressed storage to save memory space and improve computational efficiency. By identifying the absolute sparse patterns and pruning the corresponding nodes from the sequential patterns tree, the \( UG-Miner \) can improve the accuracy of the sequential pattern mining. The efficiency of \( UG-Miner \) does not change very much with the increase in the size of data set.

Figure 5 shows the comparison of time consuming between \( UG-Miner \) and \( SUF-Growth \) with the increase in the size of testing data set. In figure 5, the time consuming of \( UG-Miner \) is smaller than that of \( SUF-Growth \), because the hash compressed sketch can reduce the statistics time. By using the sketch to reduce the data size and pruning the absolute sparse patterns, the time-consuming of \( UG-Miner \) are optimized.

Figure 6 shows the comparison of memory space consuming between \( UG-Miner \) and \( SUF-Growth \) with the increase in the size of objects set. In figure 6, the memory space consuming of \( UG-Miner \) is smaller than that of \( SUF-Growth \), because the \( UG-Miner \) adopt sequential sketches and hash method to compress storage space. And by identifying the absolute sparse patterns and pruning the corresponding nodes from the sequential
patterns tree, periodically, as the data continuous arriving, the memory consumption of the UG-Miner increases slowly.

Experiment results show that the UG-Miner algorithm can save memory space and improve efficiency.

VI. CONCLUSION

This paper has addressed a new efficient sequential pattern mining algorithm for uncertain time series streams. First, for reducing memory consumption, we construct sequential sketches and pattern arrays to store continuous arriving granules of uncertain time series data. For the continuous averaging and massive size of the uncertain data streams, the sequential sketches may be saturated. Therefore, we design an optimization strategy to delete the absolute sparse patterns. And then, based on the sequential sketch, the UG-Miner algorithm for sequential pattern mining uncertain time series streams is given. Experiment results show that the UG-Miner algorithm can effectively mine frequent patterns in uncertain data streams. In future work, we plan to design distributed sequential sketches and pattern mining algorithm for distributed uncertain streams.

ACKNOWLEDGMENT

The paper is supported by the Fundamental Research Funds for the Central Universities.

REFERENCES


[8] Jiawei Han, Jian Pei, Yiwen Yin, “Mining frequent patterns without candidate generation”, In Proceedings of the 2000 ACM SIGMOD, 2000, pp. 1-12.


[19] Nishad Manerikar, and Themis Palpanas, “Frequent items in streaming data: An experimental evaluation of the state-


