International Journal of Innovative
Computing, Information and Control
Volume **8**, Number **7(A)**, July **2012**

ICIC International ©2012 ISSN 1349-4198
pp. **4675–4690**

AN EFFICIENT ONE-PASS METHOD FOR DISCOVERING BASES OF RECENTLY FREQUENT EPISODES OVER ONLINE DATA STREAMS

Min Gan and Honghua Dai

School of Information Technology
Deakin University
Burwood, VIC 3125, Australia
{ min.gan.au; honghua.dai }@gmail.com

Received November 2010; revised April 2011

Abstract. *The knowledge embedded in an online data stream is likely to change over time due to the dynamic evolution of the stream. Consequently, in frequent episode mining over an online stream, frequent episodes should be adaptively extracted from recently generated stream segments instead of the whole stream. However, almost all existing frequent episode mining approaches find episodes frequently occurring over the whole sequence. This paper proposes and investigates a new problem: online mining of recently frequent episodes over data streams. In order to meet strict requirements of stream mining such as one-scan, adaptive result update and instant result return, we choose a novel frequency metric and define a highly condensed set called the base of recently frequent episodes. We then introduce a one-pass method for mining bases of recently frequent episodes. Experimental results show that the proposed method is capable of finding bases of recently frequent episodes quickly and adaptively. The proposed method outperforms the previous approaches with the advantages of one-pass, instant result update and return, more condensed resulting sets and less space usage.*
Keywords: Data streams, Recently frequent episodes, Online mining

1. **Introduction.** The episode introduced by Mannila et al. [1] is an important pattern for modelling the relative order of occurrence of data elements over a single data sequence. For instance, the order 'A occurs before C' can be represented as a serial episode denoted as $\langle AC \rangle$. Frequent episode (FE) mining [1] is to discover episodes with supports (occurrence frequencies) no less than a user-specified threshold min_sup. FE mining is important as FEs are able to model 'common features' of a relative order of occurrences within the sequence. Since FE mining [1] was introduced, a number of mining approaches [1-7] have been proposed. The discovered FEs have been applied to many areas, such as telecommunication alarm management [1, 8], intrusion detection [9], discovery of relation between financial events and stock trends [10] and gene analysis [11].

All existing approaches calculate the frequencies of episodes in a whole non-streaming sequence [1-3, 6, 7] (or in a whole stream arrived so far [4, 5]) and find globally frequent episodes (GFEs), i.e., the episodes frequently occurring in the whole sequence (or the whole stream arrived so far). So, GFEs can be used for static analysis of non-streaming sequences and off-line data streams. However, online and dynamic analysis is needed for many online data streams like an online stream of HTTP requests received by a Web server, since the streams change over time. Over an online data stream, new data elements are generated and appended continuously and rapidly. Thus, only the recently generated stream segment may reflect the latest information and recent trends of the stream, and old data elements before a recent time point may become obsolete. It is clear that GFEs found from the whole stream cannot reflect the dynamic changes and recent trends. Therefore,

for an online data stream, FEs should be dynamically extracted from the recent stream segments instead of the whole stream. Frequent episodes found from the recent stream segment are called recently frequent episodes (RFEs).

We identify that three major problems may be caused if GFEs are used for online streams. First, no or only a small number of GFEs may be found from an online stream, since few episodes may frequently recur in the long lifetime of the stream. Second, dynamic changes and recent trends of the stream cannot be detected if GFEs are used. Due to the dynamic evolution of the stream, frequencies of episodes may change over time, and the frequency of an episode may vary dramatically in different time periods. For example, given a sample stream in Figure 1 and $\min\_sup = 2$, consider two recent time periods T_1 and T_4 with respect to current timestamps 5 and 8 respectively. Intuitively $\langle AC \rangle$ frequently occurs in T_1 and never appears in T_4; while $\langle XY \rangle$ never appears in T_1 and frequently occurs in T_4. We see that RFE $\langle AC \rangle$ reflects the recent regularities of relative order of occurrence when current time $ct = 5$, and RFE $\langle XY \rangle$ reflects the recent regularities when $ct = 8$. Nevertheless, if they are treated as GFEs, $\langle AC \rangle$ and $\langle XY \rangle$ have no difference because we only know that both of them are frequent over the stream. Third, significant FEs may be missed when they are evaluated globally. Assume that, for a stream of HTTP requests received by a Web server, a set of episodes capture common features of HTTP requests created by a kind of attack. These episodes may frequently recur only in the time periods during which the attack activity occurs; whereas over the whole stream, their frequencies may be extremely low. In this situation, the FEs corresponding to the attack cannot be detected if they are treated as GFEs. If RFEs are used, these FEs could be detected once the attack has been conducted for a certain period of time. To summarise, GFEs cannot capture the dynamic changes and recent trends of the stream; while RFEs can. Therefore, RFEs should be discovered from online data streams. Can existing GFE mining approaches [1-7] be used to discover RFEs? Although RFEs can be found by conducting any GFE mining approach on the recent stream segment whenever a new data element arrives, real-time response is hard to achieve due to the relatively long time spent on the scan and mining processes. Consequently, GFE mining approaches are not applicable for online mining of RFEs over data streams.

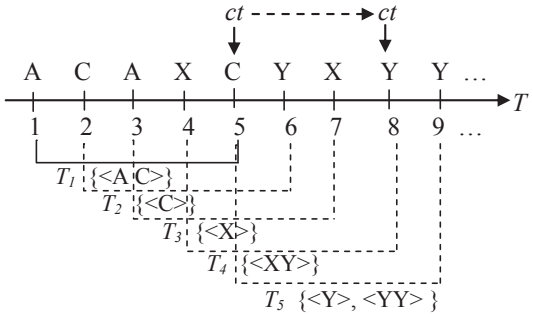


FIGURE 1. A sample stream

In this paper, we investigate online mining of RFEs over data streams. Given an online data stream, we monitor the stream online, and discover FEs frequently occurring over the most recent stream segment of length N (user-specified) instead of the whole stream whenever a new data element arrives. In contrast to traditional GFE mining, online RFE discovery raises more challenging requirements, such as one-scan of the stream, adaptive result update and instant result return. This paper aims to propose an RFE mining method satisfying these requirements. To achieve one-pass and rapid processing, three

key issues need to be considered and solved. The first is frequency measurement. In the existing frequency metrics [1, 2, 4-6], we choose $T\text{-}freq$ [4] to measure the frequencies of episodes, since it is not only anti-monotonic but also convenient for incremental computation. The second issue is condensation of RFE sets. On the one hand, the RFE sets to be found should be sufficiently condensed to avoid generating too many candidates and achieve fast processing. On the other hand, the resulting sets should be exact or approximate within a guaranteed maximum error bound. We define a highly condensed set, RFE-base, to represent the complete FE set. The third issue is the one-pass mining framework. We abstract the mining problem as two basic procedures, GFE-B-Update and RFE-B-Deduction, and construct a one-pass mining framework based on the two procedures.

The main contributions of this paper are summarised as follows.

1. We propose a new problem, on-line RFE mining over data streams.
2. A highly condensed RFE set called RFE-base is defined.
3. An efficient, one-pass method is proposed for discovering RFE-bases.
4. Through extensive experiments, we demonstrate the effectiveness and efficiency of the proposed method, and its advantages against the GFE mining approaches.

The rest of this paper is organised as follows. Section 2 reviews related works and analyses their limitations. Section 3 presents preliminaries, frequency metric $T\text{-}freq$ and problem statement. In Section 4, we introduce theorems and computing strategies for GFE-B-Update and RFE-B-Deduction. Section 5 proposes the mining framework. Experimental results are presented in Section 6 and Section 7 concludes the paper.

2. **Related Work.**

2.1. **Literature review.**

Existing GFE mining approaches can be divided into four categories.

The first is the original framework proposed by Mannila et al. [1-3]. Mannila et al. [1] introduced episodes and the FE mining problem. In [1], the support of an episode is defined as the number of sliding windows with a user-specified fixed width that contain the episode. The authors proposed an original mining framework [1], which adopts the candidate-generation-and-test strategy in frequent pattern mining [12]. Later Mannila et al. improved the original framework by introducing a new frequency metric, minimal-occurrence, and more efficient search strategies [2, 3].

The second is online stream mining methods [4, 5]. Iwanuma et al. [4] introduced a novel frequency metric, $T\text{-}freq$ for episodes, and developed an algorithm for discovering maximal frequent episodes. An FE is maximal if it has no frequent supper-episodes [4]. In [5], Laxman et al. introduced a new frequency metric, non-overlapped-occurrence. The support is defined as the number of non-overlapped occurrences sharing no common timestamps. An automaton is built for each episode. Every automaton involves waiting functions that indicate the next element types to appear. Based on the support and automatons, an algorithm was proposed for finding GFEs. Both the approaches [4, 5] need only one scan of the stream.

The third is the closed episode mining approach [7]. Recently, Zhou et al. adopted minimal-occurrence as the support and proposed an algorithm, *Clo-episode* [7] for the discovery of closed FEs, namely the FEs with no supper-episodes of the same support.

The fourth is the complex sequence mining method [6]. Huang et al. proposed a method, $EMMA$ [6] for mining GFEs from complex sequences, i.e., sequences of itemsets. The difference between $EMMA$ [6] and the above approaches is that other approaches only process simple sequences in which each data element is limited to a single item.

2.2. **A comparison between our method and previous approaches.** This section addresses the unique features of the proposed approach compared with previous approaches. We also identify the limitations of previous approaches and explore why they are not suitable for on-line RFE mining.

Table 1 outlines the major differences between previous approaches and our method. Column 2 represents whether the frequency metric adopted is anti-monotonic.

TABLE 1. A comparison between previous works and this paper

Paper	Freq. (A?)	Condensation	Dynamics	Number of scan
[1]	N	complete FE set	global	n
[2, 3]	Y	complete FE set	global	n
[4]	Y	maximal FE set	global	1
[5]	Y	complete FE set	global	1
[6]	N	complete FE set	global	n
[7]	Y	closed FE set	global	n
this paper	Y	RFE base	recent	1

2.2.1. *Dynamics of resulting sets.* This paper aims at online mining of RFEs, while previous approaches discover GFEs from a whole sequence. As addressed in Section 1, previous approaches are only suitable for static analysis of sequences as GFEs are not able to reflect the recent trends and dynamics of sequence/streams. In contrast, the approach in this paper has wider applications since it applies to both online mining of RFEs over steams and mining of GFEs over sequences.

2.2.2. *The number of scans.* As shown in Table 1, the approaches [1-3, 6, 7] need multiple scans of the sequence. Although the two approaches in [4, 5] need only one scan of the sequence in GFE mining, they still need multiple scans of the stream to discover RFEs as one additional scan of the new recent stream segment is needed whenever a new data element arrives. Hence, the existing GFE mining approaches [1-7] are not suitable for online RFE mining. In contrast, the approach in this paper aims at one-pass online mining of RFEs.

2.2.3. *Condensation of resulting sets.* Existing GFE mining approaches find three kinds of FE sets: complete FE sets [1-3, 5, 6], closed FE sets [7] and maximal FE sets [4]. A complete FE set is the set of all FEs. Complete FE sets [1-3, 5, 6] are exact but not condensed. Closed sets [13] are not sufficiently condensed [15]. Although maximal frequent pattern sets [4, 14] are highly condensed, they are information incomplete without the information of non-maximal FEs. RFE bases to be discovered in our approach will be more condensed than complete sets and closed sets.

2.2.4. *Frequency measurement.* In frequent pattern mining [12], it is suggested that anti-monotonicity is a common principle to be obeyed by any frequency metric. A frequency metric is anti-monotonic if under the frequency metric, for any pattern P and any of its supper-pattern P', the frequency of P is no less than the frequency of P' [12]. Under an anti-monotonic frequency metric, a pattern can be safely pruned if any of its sub-patterns is infrequent, and an infrequent pattern does not need to be extended (downward pruning [12]). On the contrary, if a frequency metric is not anti-monotonic, frequent patterns may be missed when the downward pruning is conducted.

To date, several typical frequency metrics for episodes [1, 2, 4-6] have been introduced. We have analysed these metrics in [16], and explored their impacts on knowledge discovery

in single sequences in [21]. The metrics used in [1, 6] do not satisfy anti-monotonicity. In the three anti-monotonic frequency metrics [2, 4, 5], we adopt $T\text{-}freq$ [4] since it is not only anti-monotonic but also convenient for incremental computation. This contributes to overtime and incremental online mining.

3. Preliminaries, Frequency Measurement and Problem Statement.

3.1. Preliminaries.

This section presents ordinary terminologies in FE mining [1].

Definition 3.1 (Data Stream). *Let I be a finite set of items, where each item denotes a distinct type of data element. A data stream S defined over I is an unbounded ordered list of data elements continuously arriving at a rapid rate, denoted as $S = (e_1)_1(e_2)_2\ldots$, where each data element is identified by both its type $e_j \in I$ and its timestamp $j \in \{1, 2, \ldots\}$. The stream segment arrived so far is denoted as $S = (e_1)_1(e_2)_2\ldots(e_{ct})_{ct}$, where ct is the current timestamp.*

Definition 3.2 (Sliding Window). *Given stream $S = (e_1)_1(e_2)_2\ldots(e_{ct})_{ct}$, a sliding window with width w over S from starting timestamp st, denoted as $win(S, st, w)$, is a stream segment defined as*

$$win(S, st, w) = \begin{cases} (e_{st})_{st}(e_{st+1})_{st+1}\ldots(e_{st+w-1})_{st+w-1} & \text{if } st + w - 1 \leq ct \\ (e_{st})_{st}(e_{st+1})_{st+1}\ldots(e_{ct})_{ct} & \text{otherwise} \end{cases} \quad (1)$$

Episodes can de divided into three classes: serial episodes, parallel episodes and composite episodes [1]. In this paper we only consider the basic type, serial episodes.

Definition 3.3 (Serial Episode). *A serial episode α over I is an ordered list of types of data elements, denoted as $\alpha = \langle a_1 a_2 a_m\rangle$, where $a_j \in I$ $(j = 1, 2, \ldots, m)$. The length of α, denoted as $\alpha.L$, is defined as m.*

In the rest of the paper, episodes are referred to as serial episodes. An episode $\alpha = \langle a_1 a_2 \ldots a_m\rangle$ is a sub-episode of another episode $\beta = \langle b_1 b_2 \ldots b_n\rangle$, denoted as $\alpha \sqsubseteq \beta$, if there exist $1 \leq i_1 < i_2 < \ldots < i_m \leq n$ such that $a_j = b_{i_j}$ for all $j = 1, 2, \ldots, m$, e.g., $\langle AB\rangle \sqsubseteq \langle ACB\rangle$.

Definition 3.4 (Occurrences of Episodes). *Given stream segment $S = (e_1)_1(e_2)_2\ldots(e_{ct})_{ct}$ and $\alpha = \langle a_1 a_2 \ldots a_m\rangle$, we say α occurs in S, and $o = (e_{i_1})_{i_1}(e_{i_2})_{i_2}\ldots(e_{i_m})_{i_m}$ is an occurrence of α in S, if there exist $1 \leq i_1 < i_2 < \ldots < i_m \leq ct$ such that $a_j = e_{i_j}$ for all $j = 1, 2, \ldots, m$. For simplicity, timestamp list $\langle i_1, i_2, \ldots, i_m\rangle$ is used to denote occurrence o. The set of occurrences of α in S is denoted as $O(S, \alpha)$.*

For example, for S in Figure 1, $O(S, \langle AC\rangle) = \{\langle 1, 2\rangle, \langle 1, 5\rangle, \langle 3, 5\rangle\}$. In addition, we say $win(S, st, w)$ contains α if α occurs in $win(S, st, w)$.

Definition 3.5 (Stream/Episode Expansion). *Given stream $S = (e_1)_1(e_2)_2\ldots(e_{ct})_{ct}$, when a new data element ae arrives, S is expanded to $(e_1)_1(e_2)_2\ldots(e_{ct})_{ct}(ae)_{ct+1}$, which is called the expansion of S with ae, denoted as $S \circ ae$. Similarly the expansion of episode $\alpha = \langle a_1 a_2 \ldots a_m\rangle$ with item b is defined as $\alpha \circ b = \langle a_1 a_2 \ldots a_m b\rangle$.*

3.2. Frequency metric.

This section reviews frequency metric $T\text{-}freq$ [4] adopted in this paper.

Definition 3.6 (Head Frequency). *Head frequency* [4] *of episode $\alpha = \langle a_1 a_2 \ldots a_m\rangle$ over S with window width w, denoted as $H\text{-}freq(S, \alpha, w)$, is defined as*

$$H\text{-}freq(S, \alpha, w) = \sum_{i=1}^{ct} \delta(win(S, i, w), \alpha) \quad (2)$$

where $\delta(win(S,i,w),\alpha)=1$ *if* $e_i=a_1$ *and* $win(S,i,w)$ *contains* α*, otherwise* $\delta(win(S,i,w),\alpha)=0$.

For instance, in Figure 1, $H\text{-}freq(S,\langle XY\rangle,3)=2$ because two windows $w(S,4,3)$ and $w(S,7,3)$ contain $\langle XY\rangle$.

Definition 3.7 (Total Frequency)**.** *Total frequency* [4] *of episode* α *over* S *with window width* w*, denoted as* $T\text{-}freq(S,\alpha,w)$*, is defined as*

$$T\text{-}freq(S,\alpha,w)=\min_{\beta\sqsubseteq\alpha} H\text{-}freq(S,\beta,w) \tag{3}$$

For example, given S in Figure 1, $T\text{-}freq(S,\langle XY\rangle,3)=2$ because $H\text{-}freq(S,\langle XY\rangle,3)=2$, $H\text{-}freq(S,\langle X\rangle,3)=2$ and $H\text{-}freq(S,\langle Y\rangle,3)=3$. $T\text{-}freq$ has a basic property below.

Proposition 3.1. *Given* S*,* $\alpha=\langle a_1a_2\dots a_m\rangle$ *and window width* w*, we have*

$$T\text{-}freq(S,\alpha,w)=\min_{i=1,2,\dots,m} H\text{-}freq(S,suf(\alpha,i),w) \tag{4}$$

where $suf(\alpha,i)=\langle a_ia_{i+1}\dots a_m\rangle$ [4].

In this paper, we adopt $T\text{-}freq$ without the window-width constraint w for the convenience of incremental mining. The $T\text{-}freq$ without w, denoted as $T\text{-}freq(S,\alpha)$, is defined as $\min_{i=1,2,\dots,m} H\text{-}freq(S,suf(\alpha,i))$. In this paper, we define $sup(S,\alpha)$ as $T\text{-}freq(S,\alpha)$.

Definition 3.8 (Frequent Episode, Maximal Frequent Episode)**.** *Episode* α *is frequent over* S *with respect to* min_*sup if* $sup(S,\alpha)\geq$ min_*sup. Episode* α *is a maximal frequent episode over* S *with respect to* min_*sup if there exist no* $\beta\sqsupseteq\alpha$ *such that* β *is frequent over* S *with respect to* min_*sup.*

In (4), all landmarks $\{i\}$ that minimise $H\text{-}freq(S,suf(\alpha,i))$, denoted as $i\text{-}\min H(S,\alpha)$, is defined as

$$\left\{i|H\text{-}freq(S,suf(\alpha,i))=T\text{-}freq(S,\alpha)=\min_{i=1,2,\dots,m} H\text{-}freq(S,suf(\alpha,i))\right\} \tag{5}$$

For example, in Figure 1, given $S=(C)_5(Y)_6(X)_7(Y)_8(Y)_9$ in T_5, $\alpha=\langle XY\rangle$, we have $H\text{-}freq(S,suf(\alpha,1))=H\text{-}freq(S,\langle XY\rangle)=1$ and $H\text{-}freq(S,suf(\alpha,2))=H\text{-}freq(S,\langle Y\rangle)=3$. Hence, $T\text{-}freq(S,\alpha)=\min\{1,3\}=1$ and $i\text{-}\min H(S,\alpha)=\{1\}$.

The $T\text{-}freq$ has an incremental property as follows.

Proposition 3.2 (Incremental Property)**.** *Given stream* $S=(e_0)_0(e_1)_1\dots(e_{ct})_{ct}$ *(*S *is empty when* $ct=0$*) and* min_*sup, let* α *be a maximal FE over* S *(let* α *be an empty episode when* S *is empty). For* $\forall ae\in I$*, if* ae *is infrequent in* $S''=S\circ ae$*,* α *is still a maximal FE over* S''*; otherwise, we have (1)* $\alpha\circ ae$ *is a maximal FE in* S''*, and (2)* $T\text{-}freq(S'',\alpha\circ ae)=\min(T\text{-}freq(S,\alpha),T\text{-}freq(S'',\langle ae\rangle))$ [4].

For example, given $S=(A)_1(C)_2(A)_3(X)_4$ and min_$sup=2$, $\alpha=\langle A\rangle$ is a maximal FE over S. When a new element $ae=C$ is appended, S becomes $S''=S\circ ae=(A)_1(C)_2(A)_3(X)_4(C)_5$. Since $\langle C\rangle$ is frequent over S'', we have $\alpha\circ ae=\langle AC\rangle$ is a maximal FE in S'', and $T\text{-}freq(S'',\langle AC\rangle)=\min(2,2)=2$. Please refer to [4] for the proofs of Propositions 3.1 and 3.2.

3.3. **Problem statement.** The problem considered in this paper is online mining of RFE-bases over data steams. To begin with we define the RFE-base which is similar to the base of frequent items introduced in [15].

Given stream segment S ($S.L=N$) and min_*sup*, the possible values of support of all FEs are in $[\min\_sup, S.L]$. If an error of support within k is tolerant, we can divide $[\min\_sup, S.L]$ into n_level sub-intervals $[\min\_sup, \min\_sup+k]$, $[\min\_sup+k+$

$1, \min\_sup + 2k + 1], \ldots, [\min\_sup + (n\_level - 1)(k + 1), S.L]$, where the size of each sub-interval is k. We set the lower bound of the i-th sub-interval as $\min\_sup_i$, and define MF_i as the maximal frequent episodes with respect to $\min\_sup_i$. For any episode α, if α is between MF_j and MF_{j+1}, then $sup(\alpha) \in [\min\_sup_j, \min\_sup_j + k]$. The collection of all MF_i $(i = 1, 2, \ldots, n\_level)$ is called an FE-base. Thus, we can use the FE base to approximately represent the complete FE set within error bound k. FE-base is formally defined as follows. Define the number of levels, n_level, as

$$n\_level = \left\lceil \frac{S.L + 1 - \min\_sup}{k + 1} \right\rceil \tag{6}$$

Let $\min\_sup_i = \min\_sup + (i - 1)(k + 1)$ $(1 \leq i \leq n\_level)$. Then the RFE-base over S with respect to $\min\_sup$ and k is defined as

$$\mathcal{B} = \cup_{i=1}^{n\_level} MF_i \tag{7}$$

For example, given S in Figure 1, $\min\_sup = 2$ and $k = 1$, the support interval [2, 9] is divided into 4 (n_level) sub-intervals: [2, 3], [4, 5], [6, 7] and [8, 9].

Assume a data stream is monitored over time. The problem considered in this paper is adaptively discovering the FE-base from the most recent stream segment whenever a new data element arrives. The problem is formally stated as follows.

Definition 3.9 (Problem Statement)**.** *Given a data stream monitored over time, and parameters: user-specified length of recent stream segment, N, $\min\_sup$ and error bound k, assume current time ct is j, and FE-base $\mathcal{B}$ over the recent N-length stream segment $S = (e_{j-N+1})_{j-N+1}\ (e_{j-N+2})_{j-N+2} \ldots (e_j)_j$ $(j \geq N)$ has been found. When a new data element, e_{j+1}, arrives $(ct = j + 1)$, the problem is discovering FE-base $\mathcal{B}'$ over the new recent N-length stream segment $S' = (e_{j-N+2})_{j-N+2}\ (e_{j-N+3})_{j-N+3} \ldots (e_{j+1})_{j+1}$. The mining process should satisfy two requirements, (1) one-pass: it need not re-scan the passed stream segment S, and (2) rapid feedback: new FE-base $\mathcal{B}'$ is found rapidly.*

To meet the two requirements, we obtain $\mathcal{B}'$ by utilising the found result $\mathcal{B}$ instead of scanning and mining S' from scratch. $\mathcal{B}'$ can be obtained from $\mathcal{B}$ in two steps: (1) $\mathcal{B}$ over S is updated to $\mathcal{B}''$ over $S'' = S \circ e_{j+1}$, and (2) $\mathcal{B}'$ is deduced from $\mathcal{B}''$. Note that S in Step (1) is $(e)_1(e)_2 \ldots (e)_j$ when $j \leq N$. The two steps are defined as two basic procedures.

1. GFE-B-Update — Given the recent N-length stream segment $S = (e_{j-N+1})_{j-N+1}$ $(e_{j-N+2})_{j-N+2} \ldots (e_j)_j$ (let $S = (e)_1(e)_2 \ldots (e)_j$ when $1 \leq j \leq N$), when a new data element e_{j+1} is appended, the appended stream segment is $S'' = S \circ e_{j+1}$. GFE-B-Update updates $\mathcal{B}$ over S to $\mathcal{B}''$ over $S'' = S \circ e_{j+1}$.
2. RFE-B-Deduction — When a new data element e_{j+1} arrives, let the new recent N-length stream segment $S' = (e_{j-N+2})_{j-N+2}\ (e_{j-N+3})_{j-N+3} \ldots (e_{j+1})_{j+1}$. In RFE-B-Deduction, $\mathcal{B}'$ over S' is deduced from $\mathcal{B}''$ over S''.

As shown in Figure 2, upon the above two procedures, the mining process can be completed in two phases described as follows.

Phase 1: $(1 \leq ct \leq N)$

(0): $(ct = 0)$ $\mathcal{B} \leftarrow \emptyset$; $S \leftarrow null$;

(1): $(ct = 1)$ When e_1 arrives, $S'' \leftarrow S \circ e_{ct} = (e_1)_1$; update $\mathcal{B}$ to $\mathcal{B}''$ by GFE-B-Update; $S \leftarrow S''$; $\mathcal{B} \leftarrow \mathcal{B}''$;

(2): $(ct = 2)$ When e_2 arrives, $S'' \leftarrow S \circ e_{ct} = (e_1)_1(e_2)_2$; update $\mathcal{B}$ to $\mathcal{B}''$ by GFE-B-Update; $S \leftarrow S''$; $\mathcal{B} \leftarrow \mathcal{B}''$;

$\vdots$

(N): $(ct = N)$ When e_N arrives, $S'' \leftarrow S \circ e_{ct} = (e_1)_1(e_2)_2 \ldots (e_N)_N$; update $\mathcal{B}$ to $\mathcal{B}''$ by GFE-B-Update; $S \leftarrow S''$; $\mathcal{B} \leftarrow \mathcal{B}''$;

Phase 2: $(ct > N)$

(N+1): $(ct = N + 1)$ When e_{N+1} arrives, $S'' = (e_1)_1(e_2)_2 \dots (e_{N+1})_{N+1}$; $S' = (e_2)_2(e_3)_3 \dots (e_{N+1})_{N+1}$; update $\mathcal{B}$ to $\mathcal{B}''$ by GFE-B-Update; $\mathcal{B}'$ is deduced from $\mathcal{B}''$ by RFE-B-Deduction; $S \leftarrow S'$; $\mathcal{B} \leftarrow \mathcal{B}'$;

$\vdots$

(j): $(ct = j)$ When e_j arrives, $S'' = (e_{j-N})_{j-N}(e_{j-N+1})_{j-N+1} \dots (e_j)_j$; new recent N-length stream $S' = (e_{j-N+1})_{j-N+1}(e_{j-N+2})_{j-N+2} \dots (e_j)_j$; update $\mathcal{B}$ to $\mathcal{B}''$ by GFE-B-Update; $\mathcal{B}'$ is deduced from B'' by RFE-B-Deduction; $S \leftarrow S'$; $\mathcal{B} \leftarrow \mathcal{B}'$;

$\vdots$

(n): $(ct = n)$ When e_n arrives, the process is terminated by the user.

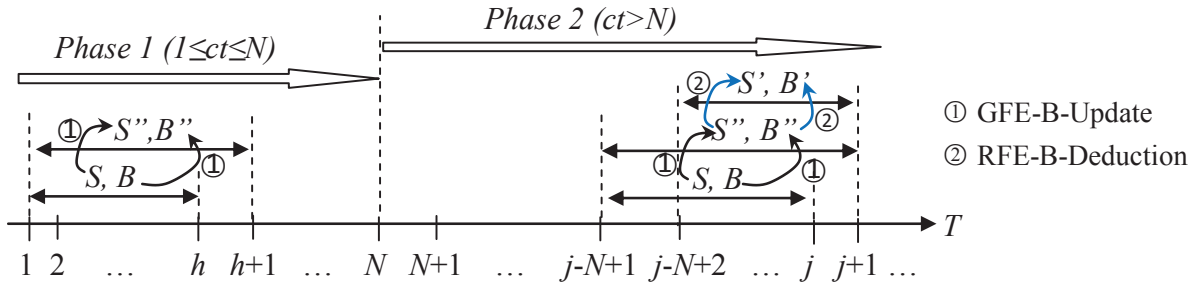


FIGURE 2. The mining process

Therefore, the mining framework can be constructed upon the the two procedures, and the mining problem can be abstracted as the two procedures.

4. **Theorems and Computing Strategies.** This section introduces theorems and computing strategies for the two procedures GFE-B-Update and RFE-B-Deduction.

4.1. **GFE-B-Update.** Since $\mathcal{B} = \cup_{i=1}^{n\_level} MF_i$, the essence of GFE-B-Update is to update MF_i to MF_i'' when a new data element e_{ct+1} arrives, where MF_i and MF_i'' respectively denote the sets of maximal FEs over $S = (e_1)_1(e_2)_2 \dots (e_{ct})_{ct}$ and $S'' = (e_1)_1(e_2)_2 \dots (e_{ct+1})_{ct+1}$ with respect to $\min\_sup_i$. From Proposition 3.2, we can deduce the incremental property of MF_i below.

Corollary 4.1 (Incremental Property of MF_i)**.** *Given N, $\min\_sup$, error bound k, sequence $S = (e_0)_0(e_1)_1 \dots (e_{ct})_{ct}$ (let $MF_i = \emptyset$ when $ct = 0$) and its corresponding appended sequence $S'' = (e_1)_1(e_2)_2 \dots (e_{ct+1})_{ct+1}$, define n_level as Equation (6). For any $i \in \{1, 2, \dots, n\_level\}$, if $T\text{-}freq(S'', \langle e_{ct+1} \rangle) < \min\_sup_i$, $MF_i'' = MF_i$; otherwise, we have*

$$MF_i'' = MF_i \times \{e_{ct+1}\} = \{\beta | \beta = \alpha \circ e_{ct+1} \wedge \alpha \in MF_i\} \tag{8}$$

$$T\text{-}freq(S'', \alpha \circ e_{ct+1}) = \min(T\text{-}freq(S, \alpha), T\text{-}freq(S'', \langle e_{ct+1} \rangle)) \tag{9}$$

The incremental property of MF_i indicates that when current S is expanded to S'' by appending a new data element e_{ct+1}, if $\langle e_{ct+1} \rangle$ is infrequent in S'', MF_i does not change in S''; otherwise MF_i'' and $T\text{-}freq$ can be obtained as Equations (8) and (9) respectively. For example, in Figure 1, given $N = 5$, $\min\_sup = 2$, error bound $k = 0$ and $S = (A)_1(C)_2(A)_3(X)_4$, assume we have obtained $MF_1 = \{A : 2\}$. When $e_5 = C$ arrives, $S'' = (A)_1(C)_2(A)_3(X)_4(C)_5$ and $T\text{-}freq(S'', \langle C \rangle) = 2 \geq \min\_sup_1 = \min\_sup = 2$. Therefore, $MF_1'' = \{\langle A \rangle\} \times \{\langle C \rangle\} = \{\langle AC \rangle\}$ and $T\text{-}freq(S'', \langle AC \rangle) = \min(2, 2) = 2$. Now consider $S = (A)_1(C)_2(A)_3(X)_4(C)_5$ and $MF_1 = \{\langle AC \rangle\}$. When $e_6 = Y$ arrives,

$S'' = (A)_1(C)_2(A)_3(X)_4(C)_5(Y)_6$. We can obtain $MF_1'' = \{\langle AC\rangle\}$ since Y is infrequent in S''.

Based on Corollary 4.1, $\mathcal{B}$ can be updated to $\mathcal{B}''$ without scanning the appended sequence S''. Furthermore, MF_i'' has the following property.

Theorem 4.1. *For any $i \in \{1, 2, \ldots, n\_level\}$, $\forall \alpha \in MF_i''$, we have (1) MF_i'' is empty or contains one and only one episode, and (2) $T\text{-}freq(S'', \alpha) = \min\_sup_i$.*

Proof: Please refer to Appendix A for the proof.

4.2. **RFE-B-Deduction.** RFE-B-Deduction is deducing $\mathcal{B}'$ from $\mathcal{B}''$. Let sequence $S'' = (e_{ct-N})_{ct-N}(e_{ct-N+1})_{ct-N+1} \ldots (e_{ct})_{ct}$ and $S' = (e_{ct-N+1})_{ct-N+1}(e_{ct-N+2})_{ct-N+2} \ldots (e_{ct})_{ct}$. S' can be generated from S'' by deleting the obsolete data element e_{ct-N}, denoted as de. Hence, the key to RFE-B-Deduction is to explore how MF_i'' is changed when de is deleted.

For any $\alpha = \langle a_1 a_2 \ldots a_m\rangle \in MF_i''$, we consider how $H\text{-}freq$ and $T\text{-}freq$ of α over S'' change when de is deleted. According to the definition of $H\text{-}freq$, if $de \neq a_1$, $H\text{-}freq$ of α does not change when de is deleted, i.e., $H\text{-}freq(S', \alpha) = H\text{-}freq(S'', \alpha)$; otherwise $H\text{-}freq(S', \alpha) = H\text{-}freq(S'', \alpha) - 1$. According to Equations (4) and (5), for $T\text{-}freq$ we need to check if there exist $j \in i\text{-}\min H(S'', \alpha)$ such that $de = a_j$. If such a j exists, $T\text{-}freq(S', \alpha) = T\text{-}freq(S'', \alpha) - 1$, else $T\text{-}freq(S', \alpha) = T\text{-}freq(S'', \alpha)$. Thus, we have the following lemma.

Lemma 4.1. *Given S'', S', $\alpha = \langle a_1 a_2 \ldots a_m\rangle \in MF_i''$, $T\text{-}freq(S'', \alpha)$ and de, we have (1) if $de \neq a_1$, then $H\text{-}freq(S', \alpha) = H\text{-}freq(S'', \alpha)$, else $H\text{-}freq(S', \alpha) = H\text{-}freq(S'', \alpha) - 1$; and (2) if $\neg\exists j \in i\text{-}\min H(S'', \alpha)$ such that $de = a_j$, then $T\text{-}freq(S', \alpha) = T\text{-}freq(S'', \alpha)$, else $T\text{-}freq(S', \alpha) = T\text{-}freq(S'', \alpha) - 1$.*

Proof: It can be proven according to the definitions of $H\text{-}freq$ and $T\text{-}freq$ straightaway.

Lemma 4.1 indicates that $T\text{-}freq$ of α decreases by 1 or does not change when de is deleted. According to Corollary 4.1, $\alpha \in MF_i'$ if $T\text{-}freq(S', \alpha) = T\text{-}freq(S'', \alpha)$; otherwise $\alpha \notin MF_i'$. If $\alpha \notin MF_i'$, a new maximal episode β with respect to $\min\_sup_i$ should be generated. This β is from the sub-episodes of α. We can prove that this β is just the sub-episode that is generated from α by deleting the j-th element (denoted as $\beta = E\text{-}delete(\alpha, j)$), where j satisfies $a_j = de$ and $j \in i\text{-}\min H(S'', \alpha)$. This conclusion is formally defined as follows.

Theorem 4.2. *Given S'', S', error bound k, MF_i'', and de, for any $\alpha = \langle a_1 a_2 \ldots a_m\rangle \in MF_i''$ ($m \geq 2$), we have $\beta \in MF_i'$ if $j \in i\text{-}\min H(S'', \alpha)$ and $de = a_j$, where $\beta = E\text{-}delete(\alpha, j) = \langle b_1 b_2 \ldots b_{m-1}\rangle = \langle a_1 a_2 \ldots a_{j-1} a_{j+1} \ldots a_m\rangle$.*

Proof: Please refer to Appendix B for the proof of the theorem.

Definition 4.1 (Computing Strategy for RFE-B-Deduction)**.** *Based on Theorem 4.2, MF_i' is deduced from MF_i'' in two major steps.*

1. *If $\neg\exists j \in i\text{-}\min H(S'', \alpha)$ such that $a_j = de$, insert α into MF_i', else do Step 2.*
2. *If $m \geq 2$ insert $\beta = E\text{-}delete(\alpha, j)$ into MF_i'.*

Then, B' can be deduced from B'' by letting $B' = \cup_{i=1}^{n\_level} MF_i'$.

Definition 4.1 indicates that when S'' is updated to S' by deleting de, if the deleted item de is in a position j of any episode $\alpha \in MF_i''$ that is in $i\text{-}\min H(S'', \alpha)$, then generate β by deleting the j-th element in α and insert β into MF_i', else let $MF_i' = MF_i''$. For example, in Figure 1, given $S'' = (A)_1(C)_2(A)_3(X)_4(C)_5(Y)_6$, $MF_1'' = \langle AC\rangle$ has been obtained. Now $de = A$ is deleted and S'' becomes $S' = (C)_2(A)_3(X)_4(C)_5(Y)_6$. For $\alpha = \langle AC\rangle \in$

MF_1'', we have $i\text{-}\min H(S'', \alpha) = \{1, 2\}$. Since there exists $j = 1 \in i\text{-}\min H(S'', \alpha)$ such that $a_1 = A = de$, we have $\beta = E\text{-}delete(\langle AC\rangle, 1) = \langle C\rangle$ and $MF_1' = \{\langle C\rangle\}$.

5. **The Mining Framework.** The mining framework can be constructed upon GFE-B-Update and RFE-B-Deduction. According to the description of the mining process in Section 3.3, the mining framework is described in Algorithm 1 (Figure 3). In the mining process, a table, FR, is used to record $T\text{-}freq$ of each item.

Algorithm 1: RFE-B-Miner(S^*, min_sup, k, N)
Input: stream $S^* = (e_1)_1(e_2)_2 \ldots$, min_sup, k and N
Output: base $\boldsymbol{B}$ found from $S = (e_{ct-N+1})_{ct-N+1}(e_{ct-N+2})_{ct-N+2} \ldots (e_{ct})_{ct}$
1: $ct \leftarrow 1$; $\boldsymbol{B} \leftarrow \emptyset$; $FR \leftarrow null$;
2: **while** $ct \leq N$ // Phase 1
3: $(\boldsymbol{B}'', FR'') \leftarrow$ *GFE-B-Update*$(\boldsymbol{B}, FR, min\_sup, n\_level, e_{ct})$;
4: $\boldsymbol{B} \leftarrow \boldsymbol{B}''$; $FR \leftarrow FR''$;
5: $ct \leftarrow ct + 1$;
6: output ($\boldsymbol{B}$);
7**: while** (not terminate) // Phase 2 (ct>N)
8: $(\boldsymbol{B}'', FR'') \leftarrow$ *GFE-B-Update*$(\boldsymbol{B}, FR, min\_sup, n\_level, e_{ct})$;
9: $(\boldsymbol{B}', FR') \leftarrow$ *RFE-B-Deduction*$(\boldsymbol{B}'', FR'', min\_sup, n\_level, e_{ct-N})$;
10: $\boldsymbol{B} \leftarrow \boldsymbol{B}'$; $FR \leftarrow FR'$;
11: output ($\boldsymbol{B}$);
12: $ct \leftarrow ct + 1$;

FIGURE 3. Algorithm RFE-B-Miner

In Algorithm 1, Phase 1 is implemented from Line 2 to Line 5. In Line 3, when new data element e_{ct} arrives, $\mathcal{B}$ and FR over $S = (e_1)_1(e_2)_2 \ldots (e_{ct-1})_{ct-1}$ are respectively updated to $\mathcal{B}''$ and FR'' over $S'' = (e_1)_1(e_2)_2 \ldots (e_{ct})_{ct}$ by calling the GFE-B-Update procedure. Phase 2 is implemented from Line 7 to Line 12. In Phase 2, when new data element e_{ct} arrives, two procedures are conducted. In Line 8, $\mathcal{B}$ and FR over $S = (e_{ct-N})_{ct-N}(e_{ct-N+1})_{ct-N+1} \ldots (e_{ct-1})_{ct-1}$ are respectively updated to $\mathcal{B}''$ and FR'' over $S'' = (e_{ct-N})_{ct-N}(e_{ct-N+1})_{ct-N+1} \ldots (e_{ct})_{ct}$ by calling the GFE-B-Update procedure. In Line 9, $\mathcal{B}'$ and FR' over $S' = (e_{ct-N+1})_{ct-N+1}(e_{ct-N+2})_{ct-N+2} \ldots (e_{ct})_{ct}$ are respectively deduced from $\mathcal{B}''$ and FR'' over $S'' = (e_{ct-N})_{ct-N}(e_{ct-N+1})_{ct-N+1} \ldots (e_{ct})_{ct}$. Since the data stream is unbounded, the algorithm is terminated by the user.

The GFE-B-Update procedure is shown in Figure 4. Its main task is to update $\mathcal{B}$ and FR over S to $\mathcal{B}''$ and FR'' over S'' respectively. The major operations (Lines 3-6) are based on Corollary 4.1.

In the RFE-B-Deduction procedure as shown in Figure 5, $\mathcal{B}'$ and FR' over S' are deduced from $\mathcal{B}''$ and FR'' over S'' respectively. According to Definition 4.1, MF_i' is deduced from MF_i'' from Line 3 to Line 8.

An example is used to illustrate how RFE-bases are discovered by RFE-B-Miner.

Example 5.1. *Given a stream as shown in Figure 1,* $\min\_sup = 2$, $k = 0$, $N = 5$, *RFE-B-Miner is used to discover RFE-bases from the stream adaptively. It terminates when* $ct = 10$.

The mining process is described as follows.

Phase 1: $(1 \leq ct \leq 5)$

Procedure 1: GFE-B-Update($\boldsymbol{B}, FR, min\_sup, n\_level, ae$)
Input: $\boldsymbol{B}, FR, min\_sup, n\_level$, and new arriving data element ae
Output: $\boldsymbol{B}''$ and FR''
1: Update FR to FR'' when ae arrives;
2: **for** i = 1 to n_level do
3: **if** in FR'', $T\text{-}freq(ae) \geq min\_sup_i$ **then**
4: $MF_i'' \leftarrow MF_i \times \{ae\}$;
5: **else**
6: $MF_i'' \leftarrow MF_i$;
7: $\boldsymbol{B}'' \leftarrow \bigcup_{i=1}^{n\_level} MF_i''$;
8: **return** $(\boldsymbol{B}'', FR'')$;

FIGURE 4. Procedure GFE-B-Update

Procedure 2: RFE-B-Deduction($\boldsymbol{B}'', FR'', min\_sup, n\_level, de$)
Input:$\boldsymbol{B}''$, FR'' min_sup, n_level and the obsolete data element to be deleted, de
Output: $\boldsymbol{B}'$ and FR'
1: Update FR'' to FR' when de is deleted;
2: **for** i = 1 to n_level do
3: Get α from MF_i'';
4: **if** $\neg\exists j \in i\text{-}minH(S'', \alpha)$ such that $a_j = de$ **then**
5: Insert α into MF_i';
6: **else if** $\alpha.L \geq 2$
7: $\beta \leftarrow E\text{-}delete(\alpha, j)$;
8: Insert β into MF_i';
9: $\boldsymbol{B}' \leftarrow \bigcup_{i=1}^{n\_level} MF_i'$;
10: **return** $(\boldsymbol{B}', FR')$;

FIGURE 5. Procedure RFE-B-Deduction

(1): $(ct = 1)$ $S'' = (A)_1$ when $e_1 = A$ arrives. $FR \leftarrow \{A : 1\}$. No frequent episodes appear. So, $\mathcal{B} \leftarrow \emptyset$; $S \leftarrow S''$.

(2): $(ct = 2)$ $S'' = (A)_1(C)_2$ when $e_2 = C$ arrives. $FR \leftarrow \{A : 1, C : 1\}$. No frequent episodes appear. So, $\mathcal{B} \leftarrow \emptyset$; $S \leftarrow S''$.

(3): $(ct = 3)$ $S'' = (A)_1(C)_2(A)_3$ when $e_3 = A$ arrives. $FR \leftarrow \{A : 2, C : 1\}$. $MF_1'' \leftarrow \{\langle A\rangle : 2\}$; $MF_1 \leftarrow MF_1''$; $\mathcal{B}'' \leftarrow \{\langle A\rangle : 2\}$; $\mathcal{B} \leftarrow \mathcal{B}''$; $S \leftarrow S''$.

(4): $(ct = 4)$ $S'' = (A)_1(C)_2(A)_3(X)_4$ when $e_4 = X$ arrives. $FR \leftarrow \{A : 2, C : 1, X : 1\}$. X is infrequent in S''. So, MF and $\mathcal{B}$ do not change; $S \leftarrow S''$.

(5): $(ct = 5)$ $S'' = (A)_1(C)_2(A)_3(X)_4(C)_5$ when $e_5 = C$ arrives. $FR \leftarrow \{A : 2, C : 2, X : 1\}$, $MF_1'' \leftarrow MF_1 \times \{C\} = \{\langle AC\rangle : 2\}$; $\mathcal{B}'' \leftarrow \{\langle AC\rangle : 2\}$; $\mathcal{B} \leftarrow \mathcal{B}''$; $S \leftarrow S''$.

Phase 2: $(ct > 5)$

(6): $(ct = 6)$

(a): (GFE-B-Update) $S'' = (A)_1(C)_2(A)_3(X)_4(C)_5(Y)_6$ when $ae = e_6 = Y$ arrives. $FR = \{A : 2, C : 2, X : 1, Y : 1\}$. Since Y is infrequent in S'', according to Corollary 4.1, $MF_1'' = MF_1 = \{\langle AC\rangle : 2\}$ and $\mathcal{B} = \{\langle AC\rangle : 2\}$.

(b): (RFE-B-Deduction) When $de = e_1 = A$ is deleted, for $\langle AC\rangle \in MF_1''$, $\exists j = 1 \in i\text{-}\min H(S'', \alpha)$ such that $a_1 = A$. So, insert $\beta = E\text{-}delete(\langle AC\rangle, 1) = \langle C\rangle$ into MF_1'. Thus, $\mathcal{B}' = MF_1' = \{\langle C\rangle : 2\}$. $\mathcal{B} \leftarrow \mathcal{B}'$.

(7): ($ct = 7$) Similarly, $\mathcal{B} = \{\langle X : 2\rangle\}$ can be found.

(8): ($ct = 8$) Similarly, $\mathcal{B} = \{\langle XY : 2\rangle\}$ can be found.

(9): ($ct = 9$) Similarly, $\mathcal{B} = \{\langle YY : 2, Y : 3\rangle\}$ can be found.

(10): Terminate.

In above, the FE-base is adaptively discovered based on GFE-B-Update and RFE-B-Deduction when the steam evolves. The obtained results at each step demonstrate the effectiveness of our approach.

6. **Experimental Results.** The proposed algorithm was performed on synthetic data. Comparisons of condensation, time efficiency and space usage were conducted between our method and three previous methods, $MINEPI$ [2], $EMMA$ [6] and $Clo\_episode$ [7]. The algorithms were implemented in Java. All experiments were performed on a computer with 2.4Ghz CPU and 1GB memory, running on windows XP.

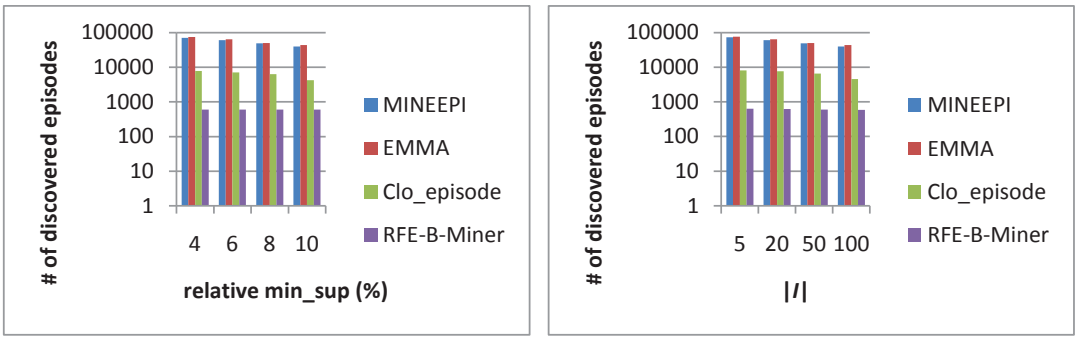
A sequence database was created by a synthetic sequence generator [18], and then the sequences in this database were connected to form a long sequence. The generation process involves three major parameters: S (k) (the number of data elements contained in S), I (the number of distinct items) and N (the length of recent sequence). We use the parameters and their values to represent a generated sequence, e.g., S10I5N5000.

6.1. **Condensation.** We performed our approach and three previous approaches on two groups of sequences: S10I20N5000 ($R\_\min\_sup$ [1] varies from 4% to 10%) and S10I5-100N5000 ($|I|$ varies from 5 to 100). The condensation of the FE sets found by different methods are measured by the number of episodes contained in each discovered set. Figure 6(a) shows an condensation comparison among the sets found from S10I20N5000 by different methods. Figure 6(b) shows an condensation comparison among the sets found from S10I5 100N5000. Note that the vertical axis of Figures 6(a) and 6(b) use logarithmic scales. From Figure 6(a), we can see the size of discovered sets by three previous approaches decreases substantially when $R\_\min\_sup$ increases. Figure 6(b) demonstrates that, for the three previous methods, the larger $|I|$ is, the less the average frequency of each episode is, and thus fewer frequent episodes are found. In contrast, the size of RFE-bases decreases slightly when relative $\min\_sup$ and $|I|$ increase. This is because its size is bounded by n_level, which is not heavily affected by relative $\min\_sup$ and $|I|$.

The results in Figure 6 demonstrate that the found RFE-bases are about 10 times smaller than the closed sets and around 100 times smaller than complete sets. Moreover, the size of the resulting sets has better scalability against $\min\_sup$ and $|I|$ than complete sets and closed sets. The high condensation makes results more concise and convenient for users to select and use. In conclusion, the results have higher condensation and better scalability.

6.2. **Time efficiency.** The time efficiency of different methods is evaluated and compared in two mining problems: GFE mining and RFE mining. GFE-mining corresponds to Phase 1 of RFE-B-Miner. Since the previous methods cannot discover RFEs incrementally, they have to scan and mine the recent N-length sequence from scratch to find current RFEs whenever a new data element arrives. Parameters $R\_\min\_sup$ and N are adjusted in the evaluation of time efficiency.

[1]Relative support and relative $\min\_sup$ ($R\_\min\_sup$) were adopted in the experiment. Relative support of episode α over S, denoted as $R\_sup(S, \alpha)$, is defined as $sup(S, \alpha)/N$, where $sup(S, \alpha) = T\text{-}freq(S, \alpha)$.



(a) Condensation vs. *min_sup* (b) Condensation vs. $|I|$

FIGURE 6. Condensation comparison

Figures 7(a) and 7(b) show the comparison of runtime on S10I20N5000 ($R\_\min\_sup$ varies from 2% to 10%) and S10I20N100-10k (N varies from 100 to 10k) respectively. For our method, the runtime of Phase 1 refers to the time spent on GFE mining over the sequence from timestamp 1 to N, and the runtime of Phase 2 refers to the average time expense on one update when a new data element arrives. The results in Figure 7 demonstrate that: (1) our method is much faster than the other three approaches in both GFE mining and RFE mining; (2) RFE-B-Miner needs only 5-10 seconds on Phase 1, and can find the new RFE-base instantly whenever a new data element arrives; and (3) the three previous approaches need around 30 to 800 seconds for the average $R\_\min\_sup$ (e.g., 6%) and N (e.g., 5000), since they have to scan and mine the recent N-length sequence from scratch. In addition, Figure 7(a) shows that, for all methods, less time is needed when $R\_\min\_sup$ increases. Figure 7(b) demonstrates that the time cost on Phase 2 is not affected by N (since RFE-Bases are computed adaptively in Phase 2); whereas the time cost of the other methods increases substantially when N increases.

The above comparison of time efficiency demonstrates that our approach is able to complete the mining within a few seconds and thus meets the real-time response requirement of stream mining. However, the previous approaches are much less efficient and not suitable for online mining of RFEs.

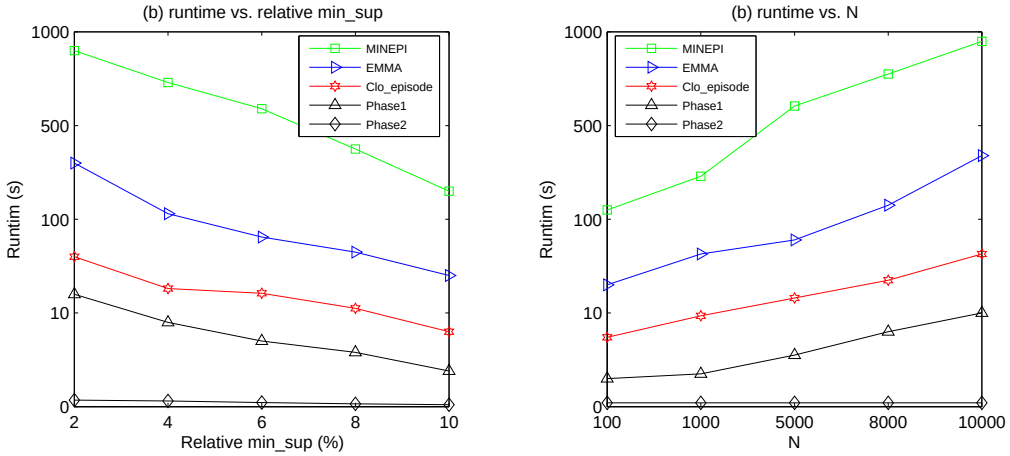
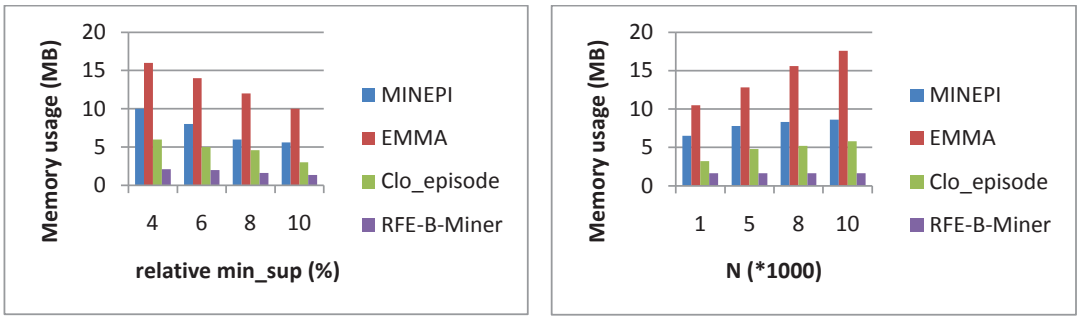


FIGURE 7. Runtime comparison

6.3. **Space usage.** Figures 8(a) and 8(b) show the comparison of space usage on S10I20N 5000 ($R\_\min\_sup$ varies from 4% to 10%) and S10I20N100-10k (N varies from 1k to 10k) respectively. The results show that RFE-B-Miner is more space efficient consuming a few mega bytes of memory. The reason why it consumes limited memory in the mining process is because the recent sequence is not kept, and only table FR, the current RFE-base, and two data elements (the added and the deleted) are recorded. As shown in Figure 8(b), the memory usage of RFE-B-Miner is not affected by N. Consequently, it can process long sequences and unbounded streams. However, the other approaches need to keep the whole sequence for mining. Hence, they consume more memory and their space usage depends on N directly. In addition, in the experiments, we found that k has little impact on space and time efficiency. So, we can specify $k = 0$ to obtain both condensed and exact resulting sets.

In summary, the results demonstrate that our approach is effective in online mining of RFEs with the advantages of condensed results and high efficiency of time and space. In contrast, previous approaches are not suitable for mining RFEs over unbounded streams.



(a) Memory usage vs. *R_min_sup* (b) Memory usage vs. N

FIGURE 8. Comparison of memory usage

7. **Conclusions and Future Work.** In this paper, we investigated online mining of recently frequent episodes over data streams. Based on frequency metric $T\text{-}freq$ [4] and FE-bases, we abstracted the mining problem as two basic procedures called GFE-B-Update and RFE-B-Deduction. An efficient one-pass method, RFE-B-Miner, was proposed for online mining of RFE-bases adaptively.

Experimental results have shown that the proposed method is capable of finding RFE-bases adaptively and rapidly with only one scan of the stream. Compared with the previous approaches, RFE-B-Miner has distinct advantages in the discovery of RFEs, such as one-pass (without tracing back passed data elements), rapid result update and return, more condensed resulting sets and less space usage.

The proposed method can be used for online mining of RFEs from various data streams such as HTTP-request streams [19]. Although the method is discussed on streams in this paper, it is also applicable to non-streaming sequences like DNA sequences [20]. Furthermore, integrating the proposed methods with fuzzy theory and proper statistical models, one could detect interesting dynamic changes in different real data streams/sequences, such as gene structure changes, HTTP-request changes and changes of real-time streaming stock quotes. These changes can be respectively applied to gene analysis, Web-access monitor and intrusion detection, and stock market analysis.

However, our approach has some deficiencies. The first is the lack of explicit restriction of window width. In our framework, the window-width restriction is underlying in the

nature distribution of episodes in the sequence. This may yield too long episodes. This is could be improved by imposing window-width restriction and adjusting GFE-B-Update and RFE-B-Deduction. The second limitation is that the proposed approach only applies to simple sequences and cannot process complex sequences [6, 21]. We aim to improve these deficiencies in the future work.

REFERENCES

[1] H. Mannila, H. Toivonen and A. Verkamo, Discovering frequent episodes in sequences, *Proc. of the 1st ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*, Montreal, Canada, pp.210-215, 1995.

[2] H. Mannila and H. Toivonen, Discovering generalized episodes using minimal occurrences, *Proc. of the 2nd ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*, Portland, OR, USA, pp.146-151, 1996.

[3] H. Mannila, H. Toivonen and A. I. Verkamo, Discovery of frequent episodes in event sequences, *Data Mining and Knowledge Discovery*, vol.1, no.3, pp.259-289, 1997.

[4] K. Iwanuma, R. Ishihara, Y. Takano and H. Nabeshima, Extracting frequent subsequences from a single long data sequence: A novel anti-monotonic metric and a simple on-line algorithm, *Proc. of the 5th IEEE Int. Conf. on Data Mining*, pp.186-193, 2005.

[5] S. Laxman, P. Sastry and K. Unnikrishnan, A fast algorithm for finding frequent episodes in event streams, *Proc. of the 13th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*, San Jose, CA, USA, pp.410-419, 2007.

[6] K. Huang and C. Chang, Efficient mining of frequent episodes from complex sequences, *Information Systems*, vol.33, no.1, pp.96-114, 2008.

[7] W. Zhou, H. Liu and H. Cheng, Mining closed episodes from event sequences efficiently, *Proc. of the 14th Pacific-Asia Conf. on Knowledge Discovery & Data Mining*, Hyderabad, India, pp.310-318, 2010.

[8] F. Bodon and Z. Hornak, Filtering false alarms: An approach based on episode mining, *Periodica Polytechnica, Electrical Engineering*, vol.49, no.1-2, pp.3-23, 2005.

[9] J. Luo and S. M. Bridges, Mining fuzzy association rules and fuzzy frequent episodes for intrusion detection, *International Journal of Intelligent Systems*, vol.15, pp.687-703, 2000.

[10] A. Ng and A. W. Fu, Mining frequent episodes for relating financial events and stock trends, *Proc. of the 7th Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, Seoul, Korea, pp.27-39, 2003.

[11] R. Bathoorn, M. Welten, M. Richardson, A. Siebes and F. J. Verbeek, Frequent episode mining to support pattern analysis in developmental biology, *Proc. of the 5th IAPR Int. Conf. on Pattern Recognition in Bioinformatics*, Nijmegen, Netherland, pp.253-263, 2010.

[12] R. Agrawal, T. Imielinski and A. Swami, Mining association rules between sets of items in large databases, *Proc. of ACM-SIGMOD the 20th Int. Conf. on Management of Data*, Washington, USA, pp.207-216, 1993.

[13] N. Pasquier, R. Bastide, R. Taouil and L. Lakhal, Discovering frequent closed itemsets for association rules, *Proc. of the 7th Int. Conf. on Database Theory*, Jerusalem, Israel, pp.398-416, 1999.

[14] R. J. Bayardo, Efficiently mining long patterns from databases, *Proc. of ACM SIGMOD the 25th Int. Conf. on Management of Data*, Seattle, Washington, USA, pp.85-93, 1998.

[15] J. Pei, G. Dong, W. Zou and J. Han, On computing condensed frequent pattern bases, *Proc. of the 2nd IEEE Int. Conf. on Data Mining*, Maebashi, Japan, pp.398-416, 2002.

[16] M. Gan and H. Dai, A study on the accuracy of frequency measures and its impact on knowledge discovery in single sequences, *Proc. of Workshops at IEEE the 10th Int. Conf. on Data Mining*, Sydney, Australia, pp.859-866, 2010.

[17] M. Gan and H. Dai, Obtaining accurate frequencies of sequential patterns over a single sequence, *ICIC Express Letters*, vol.5, no.4(B), pp.1461-1466, 2011.

[18] R. Agrawal and R. Srikant, Fast algorithms for mining association rules, *Proc. of the 20th International Conference on Very Large Data Bases*, pp.487-499, 1994.

[19] G. T. Raju, P. S. Satyanarayana and L. M. Patnaik, Knowledge discovery from web usage data: Extraction and applications of sequential and clustering patterns – A survey, *International Journal of Innovative Computing, Information and Control*, vol.4, no.2, pp.381-389, 2008.

[20] Q. Zhang, R. Zhang and B. Wang, DNA sequence sets design by particle swarm optimization algorithm, *International Journal of Innovative Computing, Information and Control*, vol.5, no.8, pp.2249-2255, 2009.

[21] M. Gan and H. Dai, Mining condensed sets of frequent episodes with more accurate frequencies from complex sequences, *International Journal of Innovative Computing, Information and Control*, vol.8, no.1(A), pp.453-470, 2012.

Appendix A. Proof of Theorem 4.1.

Proof: Proof of (1). Since initially $MF_i = \emptyset$ and Equation (8), we can deduce that MF_i'' is empty or contains one and only one episode.

Proof of (2). The basic idea is to prove that any 1-length episode α in MF_i'' satisfies $T\text{-}freq(S'', \alpha) = \min\_sup_i$, and then to prove any m-length ($m \geq 2$) episode β in MF_i'' satisfies $T\text{-}freq(S'', \beta) = \min\_sup_i$.

Given any 1-length episode $\alpha = \langle a_1 \rangle \in MF_i''$, assume $T\text{-}freq(S'', \alpha) > \min\_sup_i$. For α, we construct a supper-episode $\beta = \langle a_1 a_1 \rangle$. According to Equation (2), we have $H\text{-}freq(S'', \beta) = H\text{-}freq(S'', \alpha) - 1$. Thus, $T\text{-}freq(S'', \beta) = T\text{-}freq(S'', \alpha) - 1$ (according to Equation (3)). So, we have $T\text{-}freq(S'', \beta) > \min\_sup_i - 1$, i.e., $T\text{-}freq(S'', \beta) \geq \min\_sup_i$. That is to say a super-episode β of α is frequent with respect to $\min\_sup_i$. This contradicts the assumption of $\alpha \in MF_i''$. Therefore, for any 1-length episode α, $T\text{-}freq(S'', \alpha) = \min\_sup_i$.

Now we prove the theorem holds for any 2-length episode β in MF_i'', i.e., $T\text{-}freq(S'', \beta) = \min\_sup_i$. According to Corollary 4.1, β is generated by expanding 1-length episode α with e_{c+1}, i.e., $\beta = \alpha \circ e_{c+1}$, and $T\text{-}freq(S'', \beta) = T\text{-}freq(S \circ e_{j+1}, \beta) = \min(T\text{-}freq(S, \alpha), T\text{-}freq(S'', \langle e_{j+1} \rangle))$. Since $T\text{-}freq(S, \alpha) = \min\_sup_i$ (as proven above) and $T\text{-}freq(S'', \langle e_{j+1} \rangle) \geq \min\_sup_i$ (a condition in Corollary 4.1), $\min(T\text{-}freq(S, \alpha), T\text{-}freq(S'', \langle e_{j+1} \rangle)) = T\text{-}freq(S, \alpha) = \min\_sup_i$. Therefore, $T\text{-}freq(S'', \beta) = \min\_sup_i$. Similarly, we can prove that for any m-length ($m > 2$) episode $\gamma \in MF_i''$, $T\text{-}freq(S'', \gamma) = \min\_sup_i$.

Appendix B. Proof of Theorem 4.2.

Proof: The basic idea is to prove (1) $T\text{-}freq(S'', \beta) \geq \min\_sup_i$ and (2) if any item is inserted into β, it becomes infrequent with respect to $\min\_sup_i$.

Proof of (1). Since $\beta = E\text{-}delete(\alpha, j)$, according to Lemma 4.1, we have $T\text{-}freq(S', \beta) = T\text{-}freq(S'', \beta)$. According to anti-monotonicity, we have $T\text{-}freq(S'', \beta) \geq T\text{-}freq(S'', \alpha)$. In addition, we have $T\text{-}freq(S'', \alpha) = \min\_sup_i$ according to (2) in Theorem 4.1. Therefore, $T\text{-}freq(S', \beta) \geq T\text{-}freq(S'', \alpha) = \min\_sup_i$.

Proof of (2). We prove that $T\text{-}freq(S', \gamma) < \min\_sup_i$ if γ ($\gamma \neq \alpha$) is generated by inserting an item into β. According to anti-monotonicity, we have $T\text{-}freq(S', \gamma) \leq T\text{-}freq(S', \beta) = \min\_sup_i$. If $T\text{-}freq(S', \gamma) \geq \min\_sup_i$, then $T\text{-}freq(S'', \gamma) \geq T\text{-}freq(S', \gamma) \geq \min\_sup_i$. Since α is a maximal FE with respect to $\min\_sup_i$, γ should be a sub-episode of α. But $\gamma \not\sqsubseteq \alpha$ because $\gamma.L = \alpha.L = m$ and $\gamma \neq \alpha$. This deduces a contradiction. Therefore, $T\text{-}freq(S', \gamma) < \min\_sup_i$.