Historic Moments Discovery in Sequence Data

RAN BAI, Department of Computing, The Hong Kong Polytechnic University
WING KAI HON, Department of Computer Science, National Tsing Hua University
ERIC LO, Department of Computer Science and Engineering, Chinese University of Hong Kong
ZHIAN HE, Department of Computer Science, University of Hong Kong
KENNY ZHU, Department of Computer Science and Engineering, Shanghai Jiao Tong University

Many emerging applications are based on finding interesting subsequences from sequence data. Finding "prominent streaks," a set of the longest contiguous subsequences with values all above (or below) a certain threshold, from sequence data is one of that kind that receives much attention. Motivated from real applications, we observe that prominent streaks alone are not insightful enough but require the discovery of something we coined as "historic moments" as companions. In this article, we present an algorithm to efficiently compute historic moments from sequence data. The algorithm is *incremental* and *space optimal*, meaning that when facing new data arrival, it is able to efficiently refresh the results by keeping minimal information. Case studies show that historic moments can significantly improve the insights offered by prominent streaks alone. Furthermore, experiments show that our algorithm can outperform the baseline in both time and space.

CCS Concepts: • **Information systems → Data structures**; **Data mining**; • **Theory of computation → Data structures and algorithms for data management**;

Additional Key Words and Phrases: Historic moments, space optimal, prominent streaks, sequence data

ACM Reference format:
Ran Bai, Wing Kai Hon, Eric Lo, Zhian He, and Kenny Zhu. 2019. Historic Moments Discovery in Sequence Data. *ACM Trans. Database Syst.* 44, 1, Article 3 (January 2019), 33 pages.
https://doi.org/10.1145/3276975

1 INTRODUCTION

Finding *prominent streaks* [8], a set of maximal contiguous subsequences with values all above (or below) a certain threshold, from a sequence dataset has recently found applications in social network analysis, disease outbreak detection, and computational journalism [6, 7]. Take

This work is partly supported by the Research Grants Council of Hong Kong (GRF14200817, 15204116, 15200715, 521012), Research Committee of CUHK and NSFC grants (91646205, 61373031).
Authors' addresses: R. Bai, Department of Computing, The Hong Kong Polytechnic University; email: csrbai@comp.polyu.edu.hk; W. K. Hon, Department of Computer Science, National Tsing Hua University; email: wkhon@cs.nthu.edu.tw; E. Lo, Department of Computer Science and Engineering, Chinese University of Hong Kong; email: ericlo@cse.cuhk.edu.hk; Z. He, Department of Computer Science, University of Hong Kong; email: zahe@cs.hku.hk; K. Zhu, Department of Computer Science and Engineering, Shanghai Jiao Tong University; email: kzhu@cs.sjtu.edu.cn.

0362-5915/2019/01-ART3 $15.00
https://doi.org/10.1145/3276975

computational journalism as an example; the following news article from January 2011[1] contains a real example of a prominent streak:

> *"Today is Beijing's 36th consecutive Blue Sky Day, a day whose Air Pollution Index (API) is 100 or below, indicating "excellent" or "good" air quality. As far as I can tell, this is the longest consecutive streak of Blue Sky Days in Beijing for at least ten years."*

In the excerpt above, the prominent streak refers to a *subsequence of 36 days of consecutive measures of Air Pollution Index of 100 or below*, and the authors in [8] have developed an efficient algorithm to discover streaks like this. Although useful, a prominent streak only stands for a singular event in the dataset. In fact, the news excerpt above is continued like this:

> *" … in Beijing for at least ten years.* ***Previously, there were only three streaks of 30 days or longer, one in 2006 and two during 2008 Olympics****."*

The last sentence is crucial: it pinpoints the *rarity* of the 36-day Blue Sky streak; otherwise, readers who are unfamiliar with Beijing's weather would probably find the news mundane. In contrast, when it is further explained that the last time Beijing had 30 or more consecutive Blue Sky Days was nearly 3 years ago, readers will then be impressed by how rare the current streak is and may be aroused to learn more about Beijing's environment. In other words, the three prominent streaks in 2006 and 2008 are similar "historic moments" that happened before, which highlights the rarity of the 36-day streak that happened in 2011.

In this article, we formally introduce the concept of the *historic moment* of a streak s, which is a set of prominent streaks that end before s and can be used to highlight the *interestingness* of s. The term "interestingness" can take many forms but is mainly centered around whether a *similar* event happened before, and if so, *how long ago it was*. The technical concern is how to efficiently report historic moments from a sequence dataset, with a consideration that the data sequences are being appended regularly (e.g., hourly update of crude oil price,[2] update of the seismic magnitude per one-tenth second[3]). To this end, we present a highly efficient incremental algorithm that can enable interactive historic moment analysis on a sequence dataset with continuous data updates. The efficiency of the algorithm comes from maintaining an index in *minimal space*. Space optimality leads to disk I/O reduction per operation or even makes the index small enough to be memory resident. That property is crucial for online analysis and real-time monitoring, especially when there are possibly many data sequences of interest concurrently. Experiments on five real datasets show that our algorithm, namely, the space-optimal incremental algorithm (SOIA), outperforms the baseline algorithm by 9× to 184× in terms of speed, using 98% to 99.5% less space (e.g., in our case study, our algorithm maintained an index of only 2GB for a 350GB data sequence, whereas the baseline needed to maintain an index of size 500GB). Furthermore, our case studies show that historic moments indeed can help journalists to find full news stories, instead of half-baked ones found by prominent streaks, and help seismologists to get a bigger picture when analyzing ground motion data.

The rest of the article is organized as follows. Section 2 reviews the related work. Section 3 gives the formal problem definition. Section 4 presents our space-optimal incremental algorithm together with a few baseline algorithms. Section 5 presents the case studies, and Section 6 presents

[1]Excerpts from V. Wagner's LiveFromBeijing blog: http://www.livefrombeijing.com/2011/01/beijing-breaks-record-for-longest-streak-of-consecutive-blue-sky-days-best-air-quality-in-years/.
[2]http://www.pmbull.com/oil-price/.
[3]http://ds.iris.edu/ds/nodes/dmc/data/.

the experimental study. Section 7 concludes the article. Appendix A extends the problem and the algorithms from a single data sequence to multiple data sequences.

2 RELATED WORK

In this section, we first review related works that focus on discovering interesting knowledge from sequence datasets. The first work is [8], which studied the finding of *prominent streaks* from a data sequence $D_n = \langle v_1, v_2, \ldots, v_n \rangle$ with n numeric values. A *streak* $s = (i, j, v)$ is a *contiguous subsequence* in D_n containing numeric values $\langle v_i, v_{i+1}, \ldots, v_j \rangle$, with $i \leq j$, and $v = \min \{v_k \mid i \leq k \leq j\}$ denotes the minimum[4] value of the subsequence. We call v the *value* of s, $|s| = j - i$ the *length*[5] of s, and $[i, j]$ the *interval* of s.

Prominent streaks are in fact the two-dimensional skyline points [1, 3–5, 11, 12, 15, 18] of all the streaks found in a sequence, based on the *length* and the *value* dimensions. That is, a streak is a prominent streak if there is no streak that has both a larger length and value at the same time. The challenge of computing prominent streaks is that, given a sequence of length n, there are $\Theta(n^2)$ streaks in total. So a brute-force method would require $O(n^4)$ comparisons to locate the set of prominent streaks from $\Theta(n^2)$ candidates. In view of that, the authors in [8] developed an $O(n \log n)$ time algorithm, LLPS, which first computes a set of *local prominent streaks* ($\mathcal{LPS}_n$) from D_n and then locates the set of prominent streaks from $\mathcal{LPS}_n$. The size of $\mathcal{LPS}_n$ is at most n and it is guaranteed that the set of prominent streaks is a subset of $\mathcal{LPS}_n$. According to [8], the definition of a local prominent streak is:

Definition 2.1 (Local Prominent Streak (LPS) [8]). A streak $s = (i, j, v)$ is a *local prominent streak* if (1) $v_{i-1} < v$ and (2) $v_{j+1} < v$. We use $\mathcal{LPS}_n$ to denote the set of local prominent streaks in the sequence dataset D_n. When $i = 1$, we can ignore condition (1). Similarly when $j = n$, we can ignore condition (2).

Figure 1 shows some streaks of a sequence dataset. Streaks $s_1(299, 303, 9)$ and $s_2(306, 309, 8)$ are local prominent streaks, whereas streak $s_3(310, 311, 4)$ **is not**. The LLPS algorithm has two phases: (1) first spend $O(n)$ time to obtain the linear size $\mathcal{LPS}_n$ and (2) then invoke an $O(|\mathcal{LPS}_n| \log |\mathcal{LPS}_n|)$ skyline algorithm to compute the set of prominent streaks from $\mathcal{LPS}_n$, resulting in an overall time complexity of $O(n + |\mathcal{LPS}_n| \log |\mathcal{LPS}_n|)$, which is at most $O(n \log n)$.[6] Prominent streaks alone could only tell the continuity of a singular event. Historic moments can tell the other side of the story about how interesting that singular event is. We later show that historic moments and prominent streaks are related, but computing historic moments is a challenging problem in its own right.

In [17], the authors advocated that *rare* events are more informative and *comparisons* among objects can make a story more complete. As such, given a set $\mathcal{A}$ of concerned attributes on a relational dataset, the authors developed the concept of *top-τ-skyband* as "one of the few" objects. Intuitively, the top-τ skyband consists of (at most) τ objects that are dominated by the smallest number of other objects. Each of these objects is *one of the few* most prominent objects in the dataset when attributes in $\mathcal{A}$ are concerned. [17] proposed an efficient algorithm that finds the top-τ skyband in $O(2^d(\tau|n| + n \log n))$ time, where d is the number of attributes and n is the number of objects. Our article shares the same vision with [17] in terms of quantifying the interestingness

[4]Maximum value is an alternate but equivalent definition.
[5]An alternate definition is $|s| = j - i + 1$.
[6][8] also included another algorithm, NLPS, which generates $O(n^2)$ local prominent streaks in the first phase. It is obvious that LLPS is more efficient than NLPS.

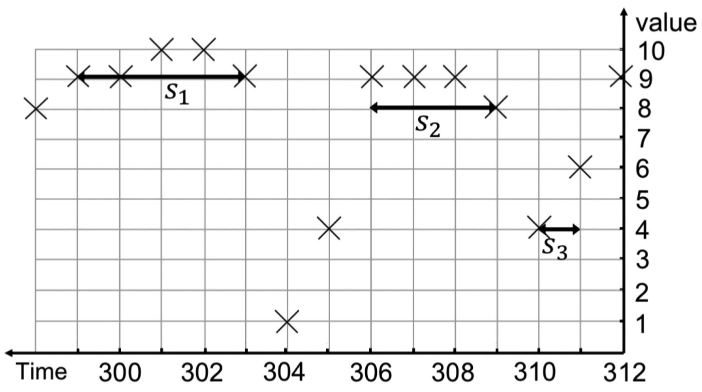


Fig. 1. A data sequence (a value is represented by ×).

of a piece of information through its rarity. However, we focus on sequence data, whereas [17] focused on relational data.

In [14], a "fact" is defined as a *contextual skyline* object that stands out against other objects in a context with regard to a set of measures. Given a relational table with a set $\mathcal{M}$ of measure attributes and a set $\mathcal{A}$ of dimension attributes, for a constraint c defined on $A \subseteq \mathcal{A}$ (known as a context) and a measure subspace $M \subseteq \mathcal{M}$, a tuple t is a *contextual skyline object* if t satisfies c and no other tuple t' satisfying c dominates t. The authors in [14] focused on identifying these "facts" in a *timely* manner. So when a new tuple t is added to a table, the target is to find which combinations of constraint c and measure subspace M make t a contextual skyline object. All those eligible $\langle c, M\rangle$ pairs are treated as *situational facts* and the *prominence* of a situational fact is defined based on the size of the skyline under c and M. Upon the arrival of a new tuple t, a baseline method can compute the topmost prominent situational facts using $O(2^{|\mathcal{M}|+|\mathcal{A}|})$ skyline queries. In case the context is fixed, the number of skyline queries becomes $O(2^{|\mathcal{M}|})$. In this article, we aim to discover certain knowledge in a timely fashion as in [14]. However, other than that, our work is orthogonal with [14] because that work focused on relational data, while ours focuses on sequence data.

All of the related work above relies on skyline computation, whose discussion began with [4], in which an $O(n^2)$ block nested loop (BNL) skyline algorithm was introduced. Then the sort-filter-skyline (SFS) algorithm was introduced [5], whose best-case time complexity is $O(dn + n \log n)$, where d is the number of dimensions. Index-based skyline algorithms were introduced in [10, 12, 15], in which the Branch & Bound (BBS) skyline algorithm in [13] gives the I/O optimal solution because for each query it visits relevant nodes of the R-tree only once, with time complexity $O(n \log n)$. Recently, [1] presented a parallel skyline algorithm. A survey of skyline computation and the variants of skyline could be found in [9].

3 PROBLEM DEFINITION

In this article, we propose the notion of *historic moment*. Our goal is to identify the historic moment in a **timely fashion**. So we mainly focus on streaks that just happened, and we name those as *situational streaks*. Furthermore, inspired by our motivating example, whether a situational streak is informative or not is relative to how long ago a similar event (streak) happened before. For some applications (e.g., computation journalism), a situational streak might lead to a news story because a similar streak happened long ago. But there are also applications (e.g., seismology) where a situational streak becomes important because a similar streak just happened. We now formally define historic moments and their related terms based on the intuition above.

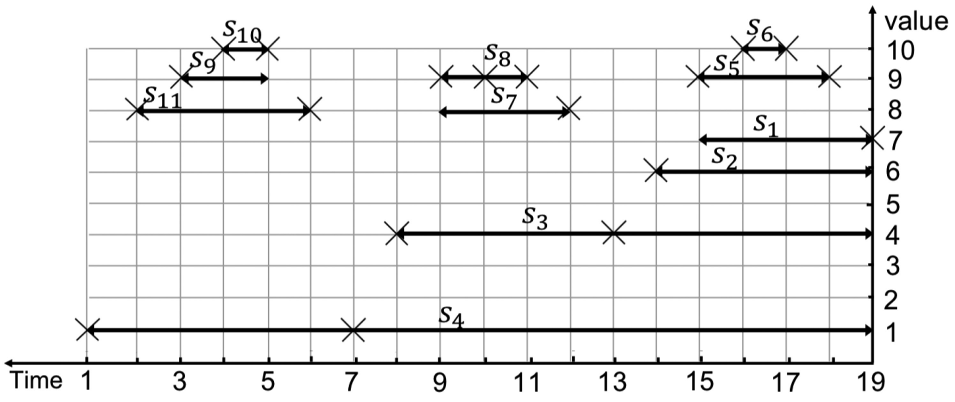


Fig. 2. Data sequence D_{19}.

Definition 3.1 (Situational Streak (SS)). In a data sequence D_n with n numeric values $\langle v_1, v_2, \ldots, v_n \rangle$, a streak $(i, j, v) \in \mathcal{LPS}_n$ is a *situational streak* if $j = n$, i.e., the most recent local prominent streak. We use $\mathcal{SS}_n$ to denote the set of situational streaks in D_n.

Figure 2 shows a data sequence D_{19} with $n = 19$ numeric values. D_{19} has four situational streaks:

$$s_1(15, 19, 7), s_2(14, 19, 6), s_3(8, 19, 4), s_4(1, 19, 1).$$

Obviously not all situational streaks are interesting, and basically we are interested in those with the highest or lowest values (e.g., high seismic magnitude, low Air Pollution Index). Without loss of generality, we focus on those with the highest values in the discussion. Also, for ease of discussion, we assume that all values are positive. So we define:

Definition 3.2 (Top-k Situational Streak). The top-k situational streaks are the k streaks in $\mathcal{SS}_n$ with the highest values.

In Figure 2, among the four situational streaks, s_1 and s_2 are the top two situational streaks.

The following news[7] is a real example of reporting not only the top but also the top two situational streaks:

> "*… the highest temperatures are above 32 Celsius for five days and 30 Celsius for six days ….*"

Next, given any situational streak z, we are interested in a subset of local prominent streaks that are "similar" to z. So we define:

Definition 3.3 (Analogous Streaks ($\mathcal{AS}$)). A local prominent streak s in a data sequence D_n is an *analogous streak* of a situational streak z when:

(1) $s.j < z.i$ (i.e., s ends before z starts),
(2) $|s| \geq |z| \cdot \sigma$ (i.e., the length of s is at least σ times that of z, where $\sigma \geq 0$ is a similarity threshold), and
(3) $s.v \geq z.v \cdot \sigma$ (i.e., the value of s is at least σ times that of z).

In Figure 2, s_{11}, with length 4 and value 8, is the only analogous streak of the top one situational streak s_1 (length 4; value 7) when the similarity threshold σ is 1. When we relax σ to be 0.75, both s_7 and s_{11} are analogous streaks of s_1.

[7] http://www.chinanews.com/sh/2015/05-21/7291066.shtml.

In this article, we use the same similarity threshold σ for both length and value. An alternate definition is to impose different similarity thresholds on length and value, which could be easily supported by straightforward adaption of our techniques.

Definition 3.4 (Historic Moments ($\mathcal{HM}$)). Let $\mathcal{AS}(z)$ be the set of analogous streaks of a situational streak z. Assume that each streak s in $\mathcal{AS}(z)$ is represented by a 3D point $(|s|, s.j, s.v)$. The *historic moment* of z, denoted by $\mathcal{HM}(z)$, is the 3D skyline of $\mathcal{AS}(z)$, and we write that as $\mathcal{HM}(z) = skyline(\mathcal{AS}(z))$.

Problem Definition: *Given a data sequence D_n with n numeric values, a similarity threshold σ, and a positive integer k, compute the historic moments for each of the top-k situational streaks.*

In Figure 2, for the top one situational streak s_1, if $\sigma = 0.5$, we have $\mathcal{HM}(s_1) = \{s_7, s_8, s_{11}\}$. Streak s_9 is not a historic moment of s_1 because s_8 *dominates*[8] s_9, denoted as $s_8 \succ s_9$, despite $\mathcal{AS}(s_1) = \{s_7, s_8, s_9, s_{11}\}$. Note that s_7 is the historic moment of s_1 with the largest j (i.e., most recent), s_8 is the historic moment of s_1 with the highest value, and s_{11} is the historic moment of s_1 with the longest interval. In the following, we use computational journalism as an example and present some real news stories that justify the use of the skyline of analogous streaks to formulate historic moments.

Story 1 (Most recent historic moment). The Beijing Blue Sky Days news in the introduction is a real example that illustrates that the most recent historic moment is newsworthy. In that story, the "36 days of Blue Sky" is a situational streak z with length 36 and value 100. The "30 days of Blue Sky that happened in 2008" is then the historic moment of z that *occurs most recently*.

Story 2 (Highest-value historic moment). In April 2014, the United Kingdom had the following news about smog:

> *"air pollution levels with more than eight lasts two days,"*

which is essentially a situational streak z of length 2 with value 8. When reporting the news, the journalist quoted the "*Great Smog*" incident, which happened in 1952:

> *"...(The 1952 Smog) led to the creation of the Clean Air Act 1956, which introduced a number of measures to reduce air pollution. ... Unfortunately in the modern day, despite the visibility and intensity of smog being much reduced, up to 29,000 people in the UK still die per year because of air pollution, according to the European Commission."*[9]

The "Great Smog" incident is in fact the historic moment of z *with the highest value.*

Story 3 (Longest-interval historic moment). In January 2014, the United States had the following news about drought in California:

> *"Meanwhile, today's scant rainfall was enough for Sacramento to finally end the longest running rainless rainy season streak in its recorded history. The city now has a new all-time record of 52 consecutive days without measurable rainy season rainfall,"*

which is essentially a situational streak of length 52 with value close to 0. When reporting the news, the journalist continued with comparisons with a historic moment that *had the longest interval*:

[8]Following the literature, an object x is said to be *dominated* by another object y, denoted as $y \succ x$, with respect to a set $\mathcal{A}$ of concerned attributes, if $y.A_i \geq x.A_i$ for all $A_i \in \mathcal{A}$, and there exists at least one attribute $A_j \in \mathcal{A}$ such that $y.A_j > x.A_j$. Here we discuss streak s, and the concerned attributes set $\mathcal{A}$ refers to $\{|s|, s.j, s.v\}$ of s.
[9]"UK Smog: You Thought This Was Bad? Take a Look at the Great Smog of 1952": http://www.independent.co.uk/news/uk/home-news/uk-smog-you-thought-this-was-bad-take-a-look-at-the-great-smog-of-1952-9238550.html.

Table 1. Major Notations in This Article

Notation	Meaning
D_n	Data sequence with n values
σ	Similarity threshold
k	Top-k parameter
$s(i,j,v)$	Streak s with interval $[i,j]$ and value v
$\mathcal{LPS}_n$	Local prominent streaks of D_n
$\mathcal{SS}_n$	Situational streaks of D_n
$\mathcal{AS}(z)$	Analogous streaks of a situational streak z
$\mathcal{HM}(z)$	Historic moments of a situational streak z
$\mathcal{P}_n$	Perplexing streaks of D_n
$\mathcal{N}_n$	Nonperplexing streaks of D_n
$\mathcal{U}_n$	Minimal subset of $\mathcal{LPS}_n$ obtained by SOIA

"... dating back to Dec. 7, 2013. The previous longest streak was Nov. 1 to Dec. 16, 1884."[10]

4 FINDING HISTORIC MOMENTS FROM A DATA SEQUENCE

In this section, we present algorithms to obtain historic moments from a data sequence. Specifically, Section 4.1 first presents a baseline algorithm that computes historic moments from a data sequence D_n offline, given a similarity threshold σ and a parameter k. In practice, there could be a data update or a user may want to look for historic moments with different σ and k values when operating under an interactive (online) mode [7]. In these cases, re-executing the baseline algorithm for any update of σ, k, or data would be inefficient. Therefore, in Section 4.2, we first present how to refactor the baseline algorithm to be *incremental*. The baseline incremental algorithm is still inefficient because it needs to keep and maintain a lot of intermediate results. Therefore, in Section 4.3, we present SOIA, an efficient incremental algorithm that returns historic moments by *maintaining and accessing minimal information*. Appendix A extends the problem and the algorithms for finding historic moments from a single data sequence to multiple data sequences. Table 1 lists the major notations used in this article.

4.1 Baseline Algorithm (BA)

Given a data sequence D_n, a similarity threshold σ, and a parameter k, the problem of finding historic moments for each of the top k situational streaks can be solved naively as follows:

- Step 1. Use the first phase of LLPS in [8] to compute the set of all local prominent streaks $\mathcal{LPS}_n$ from D_n, and then select the top k situational streaks from there. This step, according to [8], takes $O(n)$ time and the $\mathcal{LPS}_n$ takes $O(n)$ space.
- Step 2. For each top k situational streak z, scan through $\mathcal{LPS}_n$ to identify its analogous streaks $\mathcal{AS}(z)$.
- Step 3. Finally, for each z, compute the skyline from $\mathcal{AS}(z)$ as the resulting $\mathcal{HM}(z)$.

The naive method above is inefficient in terms of fully scanning the large $\mathcal{LPS}_n$ k times in Step 2. Therefore, one simple improvement is to build an index for $\mathcal{LPS}_n$ after Step 1. Specifically,

[10]"California's Devastating Drought Isn't Going to Get Better Any Time Soon": http://www.slate.com/blogs/future_tense/2014/01/30/california_s_exceptional_drought_won_t_get_better_any_time_soon.html.

one can regard each streak s in $\mathcal{LPS}_n$ as a 3D point $(|s|, s.j, s.v)$ and insert them into an R-tree. Then, the analogous streaks of a situational streak z can be regarded as a 3D-range query Q:

$$[|z| \cdot \sigma, +\infty) \times [0, z.i) \times [z.v \cdot \sigma, +\infty).$$

By building an R-tree, the above query can locate the analogous streaks of a situational streak z without scanning all the streaks in $\mathcal{LPS}_n$. Furthermore, Steps 2 and 3 can be combined as a constrained skyline query on the R-tree that returns the skyline within Q. This combined step can be implemented using the BBS skyline algorithm in [13]. Algorithm 1 summarizes the above index-based baseline algorithm (BA).

ALGORITHM 1: Baseline Algorithm (BA)

```
1: procedure BA($D_n, \sigma, k$)
2:     Use the first phase of LLPS in [8] to compute $\mathcal{LPS}_n$;
3:     Rtree = Build-R-Tree($\mathcal{LPS}_n$);
4:     $\mathcal{SS}_n$ = Get-SS($\mathcal{LPS}_n$);
5:     $\mathcal{Z}$ = Get-Top-SS($\mathcal{SS}_n$, $k$);
6:     for each streak $z$ in $\mathcal{Z}$ do
7:         $Q = [|z| \cdot \sigma, +\infty) \times [0, z.i) \times [z.v \cdot \sigma, +\infty)$; // define the analogous region
8:         $\mathcal{HM}(z)$ = BBS(Rtree, $Q$); // compute constrained skyline
```

4.2 Baseline Incremental Algorithm (BIA)

The BA is not incremental. Therefore, we refactor it to become incremental so that it can return results efficiently even when the data sequence D_n is appended with new values resulting in D_m, where $m > n$, or when the parameters are updated as σ' or k'. The incremental method is composed of two phases where there are (1) a MAINTENANCE procedure to compute $\mathcal{LPS}_m$ online when data is appended and (2) a LOOKUP procedure to answer the historic moment queries. Similar to BA, we use an R-tree to store the local prominent streaks.

4.2.1 BIA Maintenance. When the data sequence D_n is appended with new values $\langle v_{n+1}, v_{n+2}, \ldots, v_m \rangle$ resulting in D_m, where $m > n$, this procedure aims to obtain (1) the updated set of situational streaks $\mathcal{SS}_m$ and (2) possibly some new local prominent streaks. The MAINTENANCE procedure is similar to the one in [8]. Specifically, for each value $v_{n+k} \in \{v_{n+1}, v_{n+2}, \ldots, v_m\}$, where $1 \leq k \leq m - n$, it may:

(1) make some streaks in $\mathcal{SS}_{n+k-1}$ to stop being situational streaks and turn to being local prominent streaks that end at $n + k - 1$—those streaks would then get inserted into the R-tree;
(2) extend some streaks in $\mathcal{SS}_{n+k-1}$ to become longer streaks in $\mathcal{SS}_{n+k}$ that end at $n + k$; or
(3) form a new local prominent streak whose value is v_{n+k} and ends at $n + k$; this happens when none of the streak in $\mathcal{SS}_{n+k-1}$ has value v_{n+k}. Note that this is a situational streak for D_{n+k}, so it is in $\mathcal{SS}_{n+k}$.

For maintenance reasons that become clear momentarily, we separate $\mathcal{SS}_n$ from the rest of streaks in $\mathcal{LPS}_n$. Specifically, we insert streaks from $\mathcal{LPS}_n - \mathcal{SS}_n$ as 3D points into an R-tree. Streaks from $\mathcal{SS}_n \subseteq \mathcal{LPS}_n$ are stored separately. Algorithm 2 summarizes the above discussion, and we have the following:

ALGORITHM 2: BIA Maintenance Procedure

```
1: procedure MAINTENANCE($\langle v_{n+1}, v_{n+2} \dots v_m \rangle$)
2:     for $k = 1$ to $m - n$ do
3:         if $n == 0$ and $k == 1$ then
4:             $\mathcal{SS}_1 = \{(1, 1, v_1)\}$;
5:             continue;
6:         $\mathcal{SS}_{n+k} = \varnothing$;
7:         for each streak $(i, n + k - 1, v)$ in $\mathcal{SS}_{n+k-1}$ do
8:             if $v_{n+k} \geq v$ then
9:                 Insert $(i, n + k, v)$ to $\mathcal{SS}_{n+k}$;                    //case (2)
10:            else
11:                Insert $(i, n + k - 1, v)$ to Buffer $B$;                     // case (1)
12:        if no streak in $\mathcal{SS}_{n+k-1}$ has value $v_{n+k}$ then             // case (3)
13:            if all streaks in $\mathcal{SS}_{n+k-1}$ have value $< v_{n+k}$ then
14:                Insert $(n + k, n + k, v_{n+k})$ to $\mathcal{SS}_{n+k}$;
15:            else
16:                Select the streak $(i, n + k - 1, v)$ in $\mathcal{SS}_n$ whose value $v > v_{n+k}$ and is the smallest;
17:                Extend it to be $(i, n + k, v_{n+k})$
18:                Insert it into $\mathcal{SS}_{n+k}$;
19:    Insert the streaks in $B$ into R-tree;
```

- The MAINTENANCE procedure takes $\mathcal{SS}_n$ and the appending values $\langle v_{n+1}, v_{n+2}, \dots, v_m \rangle$ as the input. The outputs of it are $\mathcal{SS}_m$ and the updated R-tree by inserting new local prominent streaks.
- For each value $v_{n+k} \in \{v_{n+1}, v_{n+2}, \dots, v_m\}$, Lines 2 to 18 compute its $\mathcal{SS}_{n+k}$ and the new local prominent streaks based on $\mathcal{SS}_{n+k-1}$ iteratively. Lines 3 to 5 are to initialize $\mathcal{SS}_1$.
- For each value $v_{n+k} \in \{v_{n+1}, v_{n+2}, \dots, v_m\}$, we collect the new local prominent streaks in Buffer B (Line 11) and then insert them into the R-tree in a batch (Line 19) to reduce the I/O overhead by avoiding frequent access to the R-tree.

4.2.2 BIA Lookup. Given the R-tree of historic moment candidates created by the MAINTENANCE step, the historic moments for each of the top k situational streaks, under any similarity parameter σ' and k' values, can be obtained by calling the LOOKUP procedure as presented in Algorithm 3.

ALGORITHM 3: BIA Lookup Procedure

```
1: procedure LOOKUP($\sigma', k'$)
2:     $\mathcal{Z}$ = Get-Top-SS($\mathcal{SS}_n, k'$);
3:     for each streak $z$ in $\mathcal{Z}$ do
4:         $Q = [|z| \cdot \sigma', +\infty] \times [0, z.i) \times [z.v \cdot \sigma', +\infty]$;
5:         $\mathcal{HM}(z)$ = BBS(Rtree, $Q$); // compute constrained skyline
```

4.2.3 Space Requirement of BIA. Why is it necessary for BIA to keep all streaks in $\mathcal{LPS}_n$? Specifically, from what we have defined, the historic moments are the *skyline* of analogous streaks, which are in turn a *subset* of $\mathcal{LPS}_n$. Therefore, one might question whether BIA can keep only the skyline of $\mathcal{LPS}_n$, instead of the full $\mathcal{LPS}_n$. Unfortunately, optimizing BIA like that is incorrect. That is because a streak currently not in the $skyline(\mathcal{LPS}_n)$ could become a historic moment after a new value v_{n+1} is appended, or when facing different values of σ and k. The following are two examples.

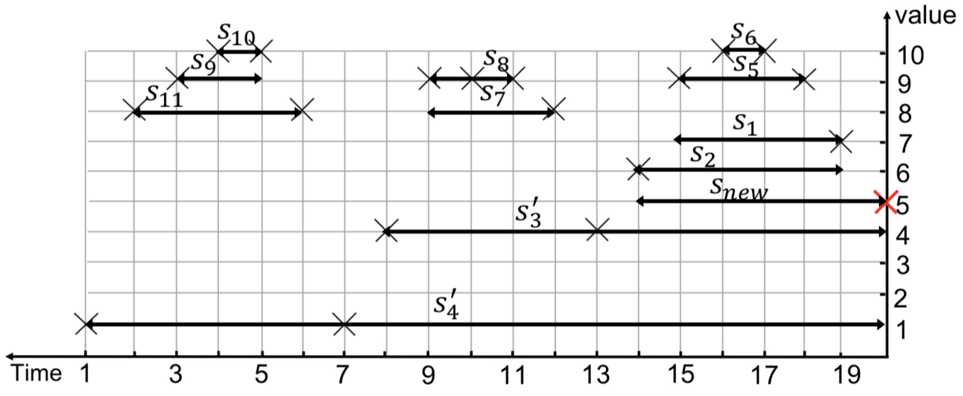


Fig. 3. After v_{20} is appended.

Example 4.1. Consider again the data sequence in Figure 2. We have:

- $n = 19$;
- $\mathcal{LPS}_{19} = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}\}$; and
- $\mathcal{SS}_{19} \subset \mathcal{LPS}_{19} = \{s_1, s_2, s_3, s_4\}$.

Consider the skyline of $\mathcal{LPS}_{19}$, which is:

- $skyline(\mathcal{LPS}_{19}) = \{s_1, s_2, s_3, s_4, s_5, s_6, s_{11}\}$.

Now consider the lookup of historic moment of the top 1 situational streak in that dataset D_{19} with $\sigma' = 0.75$. The top 1 situational streak in that dataset D_{19} is s_1, whereas the analogous streaks of s_1 are $\mathcal{AS}(s_1) = \{s_7, s_{11}\}$. The historic moments of s_1 are $\mathcal{HM}(s_1) = \{s_7, s_{11}\}$, since s_7 and s_{11} cannot dominate each other. At this point, we see that s_7 is a historic moment of s_1 but $s_7 \notin skyline(\mathcal{LPS}_{19})$.

Example 4.2. Still consider the data sequence in Figure 2, but with a new value $v_{20} = 5$ appended, resulting in D_{20} of Figure 3. We see that, with v_{20} appended, it has:

(1) made streaks s_1 and s_2 in $\mathcal{SS}_{19}$ to stop being situational streaks and turn to being local prominent streaks that end at $n = 19$;
(2) extended streaks s_3 and s_4 in $\mathcal{SS}_{19}$ to become longer streaks s'_3 and s'_4 in $\mathcal{SS}_{20}$ that ends at $n = 20$; and
(3) formed a new local prominent streak, s_{new}, whose value is 5 and ends at $n = 20$.

Now consider the lookup of historic moment of the top 1 situational streak in that updated dataset D_{20} with $\sigma' = 0.5$. The top 1 situational streak in that dataset D_{20} is s_{new}, whereas the analogous streaks of s_{new} are $\mathcal{AS}(s_{new}) = \{s_7, s_{11}\}$. The historic moments of s_{new} are $\mathcal{HM}(s_{new}) = \{s_7, s_{11}\}$. Once again, we see that s_7 is a historic moment of s_{new} but $s_7 \notin skyline(\mathcal{LPS}_{19})$.

4.3 Space-Optimal Incremental Algorithm (SOIA)

BIA needs to keep and maintain $\mathcal{LPS}_n$, which is space inefficient, especially when multiple sequences are of interest (e.g., multiple stocks, multiple seismic monitoring sensors) or when the sequences are very long (e.g., high-frequency trading with stock tick every millisecond). As shown in the previous section, straightforward optimizations like keeping $skyline(\mathcal{LPS}_n)$, instead of $\mathcal{LPS}_n$, is unfortunately incorrect. When not space efficient, BIA would not be time efficient either because the redundancies in the index would jeopardize both the lookup and maintenance time. In this section, we present a space-optimal incremental algorithm (SOIA) that keeps only a minimal subset $\mathcal{U}_n$ of $\mathcal{LPS}_n$ that is sufficient to return historic moments online. Formally,

Definition 4.3. Given a data sequence D_n, the MINIMAL SUBSET $\mathcal{U}_n$ of $\mathcal{LPS}_n$ refers to a subset of $\mathcal{LPS}_n$ that guarantees

(1) for any streak s in $\{\mathcal{LPS}_n - \mathcal{U}_n\}$, it must not be a historic moment of any situational streak y in D_m, where $m \geq n$, and
(2) there does **NOT** exist a proper subset X of $\mathcal{U}_n$ that satisfies condition (1).

SOIA strikes to maintain $\mathcal{U}_n$ under continuous data updates. Space optimality of SOIA significantly reduces the size of the index, thereby reducing the I/O per operation or making the index memory resident. That property is crucial for online analysis and real-time monitoring, especially when there are possibly many data sequences of interest, which demands one index per data sequence.

So the grand challenge of SOIA is how to confine $\mathcal{U}_n$, where trivially confining $\mathcal{U}_n$ to be the skyline of $\mathcal{LPS}_n$ is definitely insufficient, whereas confining $\mathcal{U}_n$ to be $\mathcal{LPS}_n$ is definitely not space optimal. Now we first discuss how to obtain $\mathcal{U}_n$ properly given the data sequence D_n, and Section 4.3.1 studies how to maintain its minimality when the data sequence is appended.

In SOIA, we treat σ as a parameter that controls the tradeoff between space and time, in addition to its original role as being the similarity threshold. More specifically, when deciding which streaks in $\mathcal{LPS}_n$ shall be kept (i.e., belong to $\mathcal{U}_n$) and which shall be discarded, a small σ value makes local prominent streaks easier to be an analogous streak of a situational streak z. So it is natural that the size of $\mathcal{U}_n$ increases when σ is getting smaller. In contrast, a large σ value makes local prominent streaks harder to be an analogous streak of a situational streak z. So it is intuitive that the size of $\mathcal{U}_n$ decreases when σ is getting larger. With σ as a parameter between space and time, the goal of SOIA is to maintain $\mathcal{U}_n$ as long as a user queries for historic moments with any similarity value σ' larger than or equal to σ.

We now confine what $\mathcal{U}_n$ should be, given σ. We start by showing a simple lemma (Lemma 4.4). Then, we look at the easiest case, with $\sigma = 1$. We can interpret this case as either having the most stringent space requirement or as the users being uninterested in analogous streaks that are shorter, or smaller in value, than the situational streak of interest. Finally, we work on the general case.

LEMMA 4.4. *The intervals of two local prominent streaks are either disjoint or one containing the other.*

PROOF. Assume to the contrary that there exist two local prominent streaks (i, j, v) and (i', j', v') whose intervals are not disjoint nor one containing the other. Without loss of generality, assume that $i < i'$. Then, we have $i < i' \leq j < j'$. That is, $i' - 1$ is within the range of $[i, j]$ and $j + 1$ is within the range of $[i', j']$. Now, there are two cases:

(1) If $v \geq v'$, then (i', j', v') is not a local prominent streak because $v_{i'-1} \geq v \geq v'$.
(2) Otherwise, we have $v' > v$. Then (i, j, v) is not a local prominent streak because $v_{j+1} \geq v' > v$.

So both cases lead to contradiction. The lemma follows. □

The Specific Case: What Streaks in $\mathcal{LPS}_n$ Should Be in $\mathcal{U}_n$ When $\sigma = 1$?

Let $\mathcal{HM}(\mathcal{SS}_n)$ denote the set that contains the historic moments of all streaks in $\mathcal{SS}_n$, and when $\sigma = 1$, we have the following proposition:

Proposition 4.5. *When $\sigma = 1$, $skyline(\mathcal{LPS}_n) \cup \mathcal{HM}(\mathcal{SS}_n)$ is the minimal subset $\mathcal{U}_n$ of $\mathcal{LPS}_n$ that contains historic moments of a situational streak for dataset D_n or future dataset D_m, where $m > n$.*

Proof. We prove by showing that a streak s is in $skyline(\mathcal{LPS}_n)$ or $\mathcal{HM}(\mathcal{SS}_n)$ if and only if s can serve as a historic moment of some streaks in $\mathcal{SS}_n$ or $\mathcal{SS}_m$. This is done by proving Lemma 4.6 (the if case) and Lemma 4.7 (the only-if case) below.

Lemma 4.6. *If a streak s in $\mathcal{LPS}_n$ can serve as a historic moment of a streak in $\mathcal{SS}_n$ or $\mathcal{SS}_m$, then s is in $skyline(\mathcal{LPS}_n) \cup \mathcal{HM}(\mathcal{SS}_n)$.*

Proof. It is equivalent to showing that for any streak s, if $s \notin skyline(\mathcal{LPS}_n)$ and $s \notin \mathcal{HM}(\mathcal{SS}_n)$, then s will not be a historic moment of any streak in $\mathcal{SS}_n$ or $\mathcal{SS}_m$.
Since s is not in the skyline of $\mathcal{LPS}_n$, there must exist some streak $y \in \mathcal{LPS}_n$ in the skyline with $y \succ s$. Now, for any streak $z^* = (i^*, j^*, v^*)$ in $\mathcal{SS}_n$ or $\mathcal{SS}_m$:

(1) If $z^* \in \mathcal{SS}_n$, s cannot be a historic moment of z^* since $s \notin \mathcal{HM}(\mathcal{SS}_n)$.
(2) Else, if y does not overlap with z^*, y will be an analogous streak of z^* whenever s is, so that s cannot be in the skyline of $\mathcal{AS}(z^*)$, and thus not a historic moment of z^*.
(3) Else, y overlaps with z^* and $z^* \notin \mathcal{SS}_n$. Then, we must have some $z = (i, j, v)$ in $\mathcal{SS}_n$ with the same starting position as z^*, i.e., $i = i^*$, and also y must overlap with z. Also, $|z| < |z^*|$ since $z^* \notin \mathcal{SS}_n$ while both streaks z and z^* have the same starting position. Now, recall that from Lemma 4.4, the intervals of two local prominent streaks are either disjoint or one containing the other. Thus, we further have $|s| \leq |y|$ (since $y \succ s$) and $|y| \leq |z|$ (since y overlaps with z and $z \in \mathcal{SS}_n$). The above inequalities imply that $|s| < |z^*|$ so that s is not an analogous streak, and thus not a historic moment of z^* when $\sigma = 1$.

So in all cases, s is not a historic moment of z^*. This completes the proof of the lemma. □

Lemma 4.7. *If a streak s is in $skyline(\mathcal{LPS}_n) \cup \mathcal{HM}(\mathcal{SS}_n)$, then s can serve as a historic moment of $\mathcal{SS}_n$ or $\mathcal{SS}_m$.*

Proof. Since any streak in $\mathcal{HM}(\mathcal{SS}_n)$ is already a historic moment of a current situational streak, it remains to show that any streak $s = (i, j, v)$ in $skyline(\mathcal{LPS}_n)$ can be a historic moment of some streak in $\mathcal{SS}_n$ or $\mathcal{SS}_m$. To see this, imagine that the data sequence D_n is appended with the following values for its next $|s| + 2$ days: $\epsilon, v, v, \ldots, v$, where ϵ is a positive value less than v. That essentially will create a situational streak $z^* = (i^*, j^*, v^*)$, with $i^* = n + 2$, $j^* = n + |s| + 2$, length $|z^*| = j^* - i^* = |s|$, and value $v^* = v$.

Note that s is an analogous streak of z^*. Furthermore, s is in the skyline of $\mathcal{AS}(z^*)$ because (1) every streak in $\mathcal{AS}(z^*)$ must end before $n + 1$, but (2) every streak that ends exactly at $n + 1$ will have minimum value ϵ, which cannot be an analogous streak of z^*. This implies that $\mathcal{AS}(z^*) \subseteq \mathcal{LPS}_n$. So no streak in $\mathcal{AS}(z^*)$ dominates s as $s \in skyline(\mathcal{LPS}_n)$. Thus, s is in the skyline of $\mathcal{AS}(z^*)$, and is therefore a historic moment of z^*. This completes the proof of the lemma. □

By combining Lemmas 4.6 and 4.7, Proposition 4.5 follows. □

For Figure 2, as in Example 4.1, we have:

- $\mathcal{SS}_{19} = \{s_1, s_2, s_3, s_4\} and$
- $skyline(\mathcal{LPS}_{19}) = \{s_1, s_2, s_3, s_4, s_5, s_6, s_{11}\}$.

When we set $\sigma = 1$ and $k = 4$, the historic moment of $\mathcal{SS}_{19}$, denoted as $\mathcal{HM}(\mathcal{SS}_{19})$, is $\{s_{11}\}$, since s_{11} is a historic moment (and the only one) of s_1, while there are no historic moments for s_2, s_3, or s_4. So Proposition 4.5 states that we only need to keep $\{s_1, s_2, s_3, s_4, s_5, s_6, s_{11}\}$. Although currently s_5 and s_6 are not historic moments of any situational streak (since they overlap with all streaks in $\mathcal{SS}_{19}$), each of them may be a historic moment of some situational streaks in the future. For example, consider two new values that $v_{20} = v_{21} = 9$ are added to the example data sequence in Figure 2. Then, $(20, 21, 9)$ becomes a situational streak of the data sequence D_{21}, with both s_5 and s_6 being its historic moments.

Example 4.8. Consider v_2 and v_6 equal to 7 instead of 8 in Figure 2. In this case, streak $s_{11} = (2, 6, 7)$ is no longer in $skyline(\mathcal{LPS}_{19})$, since it is dominated by s_1. Yet, s_{11} is a historic moment of s_1, and $s_{11} \in \mathcal{HM}(\mathcal{SS}_{19})$.

The example above illustrates why Proposition 4.5 has to keep $\mathcal{HM}(\mathcal{SS}_n)$ as well. In that example, $s_{11} \in \mathcal{HM}(\mathcal{SS}_{19})$. This also justified our aforementioned argument of why modifying BIA to keep only the skyline of $\mathcal{LPS}_n$ is insufficient.

The General Case: What Streaks in $\mathcal{LPS}_n$ Should Be in $\mathcal{U}_n$ When $\sigma > 0$?

Unfortunately, Proposition 4.5 is still insufficient, specifically when there are queries with $\sigma' < 1$. As we discussed in Example 4.1, in Figure 2, when querying with $\sigma' = 0.75$, s_7 is the historic moment of s_1, but $s_7 \notin skyline(\mathcal{LPS}_{19}) \cup \mathcal{HM}(\mathcal{SS}_{19})$ according to Proposition 4.5.

So to support a general σ, in addition to $skyline(\mathcal{LPS}_n) \cup \mathcal{HM}(\mathcal{SS}_n)$, what else shall we include while maintaining minimality?

We answer the question by continuing Example 4.1. Specifically, when $\sigma = 0.75$, s_7 shall be included, but unfortunately it is *pruned* by s_5 because $s_5 \succ s_7$. However, while s_7 is pruned by s_5, the latter is not serving as an analogous streak of s_1 instead. Why can't s_5 serve as s_1's analogous streak? That is because s_5 *overlaps* with s_1, which violates the definition of analogous streak (Definition 3.3, condition 1; i.e., s_5 has to end before s_1 starts). In other words, when s_5 cannot serve as an analogous streak of s_1, it shall not be used to prune any analogous streak for s_1.

So we classify the streaks in $\mathcal{LPS}_n$ into two subsets: (1) *Perplexing Streaks* $\mathcal{P}_n$, and (2) *Nonperplexing Streaks* $\mathcal{N}_n$:

Definition 4.9 (Perplexing Streaks $\mathcal{P}_n$). A streak $p \in \mathcal{LPS}_n$ is a *perplexing* streak when there exists a situational streak $z \in \mathcal{SS}_n$ such that:

(1) p overlaps with z,
(2) $|p| \geq |z| \cdot \sigma$, and
(3) $p.v \geq z.v \cdot \sigma$.

Note that p can overlap with multiple situational streaks. We use $\mathcal{P}_n$ to denote the set of all perplexing streaks in data sequence D_n.

By definition, $\mathcal{SS}_n \subseteq \mathcal{P}_n$.

Definition 4.10 (Nonperplexing Streaks $\mathcal{N}_n$). Streaks in $\mathcal{LPS}_n$ that are not in $\mathcal{P}_n$ are *nonperplexing* streaks, denoted as $\mathcal{N}_n$. That is, $\mathcal{N}_n = \mathcal{LPS}_n - \mathcal{P}_n$. So a streak $s \in \mathcal{LPS}_n$ is a nonperplexing streak if, for every situational streak $z \in \mathcal{SS}_n$ that s overlaps with, either $|s| < |z| \cdot \sigma$ or $s.v < z.v \cdot \sigma$.

Example 4.11. In Figure 2, when $\sigma = 0.75$, streaks in $\mathcal{LPS}_{19} = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}\}$ are classified into

- $\mathcal{P}_{19} = \{s_1, s_2, s_3, s_4, s_5\}$ *or*
- $\mathcal{N}_{19} = \{s_6, s_7, s_8, s_9, s_{10}, s_{11}\}$.

s_5 is in $\mathcal{P}_{19}$ because (1) s_5 overlaps with s_1, (2) $|s_5| \geq |s_1| \cdot 0.75$, and (3) $s_5.v \geq s_1.v \cdot 0.75$. In contrast, $s_6 \notin \mathcal{P}_{19}$ because $|s_6| \ngeq |s_i| \cdot 0.75$ for any $s_i \in \mathcal{SS}_{19}$.

In the following, we state some important lemmas.

Lemma 4.12. *A streak that is dominated by skyline($\mathcal{N}_n$) would not be a historic moment of any streak in either $\mathcal{SS}_n$ for dataset D_n or $\mathcal{SS}_m$ for future dataset D_m, where $m > n$.*

Proof. Suppose that streak s is dominated by a streak $y \in skyline(\mathcal{N}_n)$. Then, for any streak z^* in $\mathcal{SS}_n$ or $\mathcal{SS}_m$, two cases can happen: (1) y does not overlap with z^* or (2) y overlaps with z^*.

In Case (1), if s is an analogous streak of z^*, so is y, as $y \succ s$, so that s is not in the skyline of $\mathcal{AS}(z^*)$, and thus not a historic moment of z^*.

In Case (2), let $z^* = (i^*, j^*, v^*)$. Since y overlaps with z^*, we must have some $z = (i, j, v)$ in $\mathcal{SS}_n$ with the same starting position as z^*, i.e., $i = i^*$, and also y overlaps with z. (Note that $z = z^*$ if $z^* \in \mathcal{SS}_n$.) As $y \in \mathcal{N}$, $|y| < |z|\,\sigma \leq |z^*|\,\sigma$ holds. As $y \succ s$, this further implies $|s| \leq |y| < |z^*|\,\sigma$. Thus, s is not an analogous streak of z^*. Consequently, z cannot be a historic moment of z^*. This completes the proof. □

Lemma 4.12 directly leads to the following corollary.

Corollary 4.13. *Given a streak $s \in \mathcal{N}_n$, if $s \in \mathcal{HM}(z)$, where $z \in \mathcal{SS}_n \cup \mathcal{SS}_m$, then $s \in skyline(\mathcal{N}_n)$.*

Definition 4.14 (Smallest-Value Extension). Let $z = (i, j, v)$ be a situational streak. Suppose that the sequence is appended with a new value $v' < v$. If the streak $z' = (i, j+1, v')$ remains as a streak in $\mathcal{LPS}_{n+1}$ (i.e., $v_{i-1} < v'$), then z' is called the v' extension of z. If v_{small} is the smallest value v' such that the v' extension of z exists, then the streak $z^+ = (i, j+1, v_{small})$ is called the smallest-value extension of z.

So for Figure 2, the corresponding smallest-value extensions of the situational streaks are

$$s_1^+ = (15, 20, 6+\epsilon) \qquad s_2^+ = (14, 20, 4+\epsilon)$$
$$s_3^+ = (8, 20, 1+\epsilon) \qquad s_4^+ = (1, 20, 0+\epsilon),$$

where ϵ is an arbitrarily small positive value. Note that streak $(15, 20, 7)$ is not s_1^+ because 7 is not the smallest possible value to extend $s_1 = (15, 19, 7)$. Appending the data sequence with $v_{small} = 6$ will result in a situational streak $(14, 20, 6)$. But that does not qualify as s_1^+ because the starting position of s_1^+ should be the same as the starting position of s_1.

Definition 4.15 (Universal Domination). Let $\mathcal{SS}_n^{(p)}$ be the set of situational streaks that overlap with a perplexing streak p. A streak s is *universally dominated* by a perplexing streak p when

(1) $p \succ s$, and
(2) s is not an analogous streak of all $z \in \mathcal{SS}_n^{(p)}$, and
(3) s is not an analogous streak of z^+, for all $z \in \mathcal{SS}_n^{(p)}$.

Conceptually, universal domination is harder to achieve than the ordinary domination, since p universally dominating s implies that p dominates s.

Lemma 4.16. *If a streak $s \in skyline(\mathcal{N}_n) \cup \mathcal{P}_n$ that is either (1)* universally dominated *by some streak in $\mathcal{P}_n$ or (2) its length is less than σ, then s would not be a historic moment of any streak in $\mathcal{SS}_n$ for dataset D_n or $\mathcal{SS}_m$ for future dataset D_m, where $m > n$.*

Proof.

(1) Suppose that a streak s is universally dominated by a perplexing streak $p = (i', j', v') \in \mathcal{P}_n$. Then, there are the following cases when considering a current/future situational streak $z^* = (i^*, j^*, v^*)$:
   (a) z^* does not overlap with p. Then, if s is an analogous streak of z, so is p. In this case, s is not in the skyline of $\mathcal{AS}(z^*)$, and thus not a historic moment of z^*.
   (b) z^* overlaps with p.
      (i) $z^* \in \mathcal{SS}_n$: then $z^* \in \mathcal{SS}_n^{(p)}$, so that by definition s cannot be an analogous streak of z^*.
      (ii) $z^* \notin \mathcal{SS}_n$ (it happens when $z^* \in \mathcal{SS}_m$ and $z^* \notin \mathcal{SS}_n$): then there exists some current situational streak $z = (i, j, v)$ with the same starting position as z^*, and $z \in \mathcal{SS}_n^{(p)}$. Furthermore, $|z^*| \geq |z^+|$ and $z^*.v \geq z^+.v$ hold, where z^+ is the smallest-value extension of z. As s is not an analogous streak of z^+, s cannot be an analogous streak of z^*. In other words, s cannot be a historic moment of z^*.

(2) Any streak in $\mathcal{SS}_n$ or $\mathcal{SS}_m$ has length at least 1. So streaks in $skyline(\mathcal{N}_n) \cup \mathcal{P}_n$ with length less than σ cannot be the historic moment, as their length is not qualified.

This completes the proof. □

Lemma 4.16 leads to the following corollary.

Corollary 4.17. *Given a streak $s \in skyline(\mathcal{N}_n) \cup \mathcal{P}_n$, if $s \in \mathcal{HM}(z)$, where $z \in \mathcal{SS}_n \cup \mathcal{SS}_m$, then $\forall p \in \mathcal{P}_n$ such that s is not universally dominated by p, and s has length no less than σ.*

Example 4.18. Consider Figure 2 again; recall that when $\sigma = 0.75$, $\mathcal{P}_{19} = \{s_1, s_2, s_3, s_4, s_5\}$, and $\mathcal{N}_{19} = \{s_6, s_7, s_8, s_9, s_{10}, s_{11}\}$. Also, $skyline(\mathcal{N}_{19}) = \{s_6, s_7, s_8, s_{11}\}$. Then, s_5 is a perplexing streak. Note that $s_8 \in skyline(\mathcal{N}_{19})$ and s_8 is universally dominated by s_5 since:

(1) $s_5 \succ s_8$,
(2) s_8 is not an analogous streak of any streak in $\mathcal{SS}_{19}^{(s_5)} = \{s_1, s_2, s_3, s_4\}$, and
(3) s_8 is not an analogous streak of $s_1^+, s_2^+, s_3^+, s_4^+$.

Therefore, by Lemma 4.16, s_8 would not be a historic moment in any case and it is not necessary to keep it. In contrast, $s_7 \in skyline(\mathcal{N}_{19})$ and $s_5 \succ s_7$ as well. However, since s_7 is an analogous streak of s_1, s_7 is not universally dominated by s_5 and it cannot be pruned using Lemma 4.16.

Theorem 4.19. *The minimal subset $\mathcal{U}_n$ of $\mathcal{LPS}_n$ is the set of streaks in $skyline(\mathcal{N}_n) \cup \mathcal{P}_n$ that (1) are not universally dominated by any streak in $\mathcal{P}_n$ and (2) have length at least σ.*

Proof. We prove the theorem by showing that a streak s is in $\mathcal{U}_n$ if and only if s can serve as a historic moment of some streak in $\mathcal{SS}_n$ or $\mathcal{SS}_m$, where $m > n$. This is done by proving Lemma 4.20 (the if case) and Lemma 4.21 (the only-if case) below.

Lemma 4.20. *If a streak s can serve as a historic moment of some streak in $\mathcal{SS}_n$ or $\mathcal{SS}_m$, then s is in $\mathcal{U}_n$.*

Proof. First, $\mathcal{LPS}_n = \mathcal{N}_n \cup \mathcal{P}_n$ contains all local prominent streaks, and thus all historic moments. Then, by Corollary 4.13, we see that $skyline(\mathcal{N}_n) \cup \mathcal{P}_n$ contains all historic moments. Consequently, by Corollary 4.17, we see that $\mathcal{U}_n$ contains all historic moments. The lemma follows. □

Lemma 4.21. *If a streak s is in $\mathcal{U}_n$, then s can serve as a historic moment of some streak in $\mathcal{SS}_n$ or $\mathcal{SS}_m$.*

Proof. A streak s in $\mathcal{U}_n$ is either (1) not dominated by any other streak in $\mathcal{LPS}_n$ or (2) dominated only by some streak $p \in \mathcal{P}_n$ but not universally dominated by p.

For Case (1), if we append the data sequence D_n first with an arbitrarily small positive value ϵ, followed by $\lfloor |s|/\sigma \rfloor + 1$ values of $(s.v)/\sigma$, then, after $\lfloor |s|/\sigma \rfloor + 2$ new data values arrived, there will be a situational streak z^* with length $|z^*| = \lfloor |s|/\sigma \rfloor$ and value $z.v = (s.v)/\sigma$. That streak z^* will regard s as its historic moment.[11]

For Case (2), let z be the longest situational streak in $\mathcal{SS}_n$ such that s is an analogous streak of z^*, where $z^* = z$ or z^+. Note that such a z must exist since s is not universally dominated by some $p \in \mathcal{P}_n$ (recall Definition 4.15 for the property of universal domination). Also, $z^* \in \mathcal{SS}_n \cup \mathcal{SS}_m$.

Then, for any streak $p \in \mathcal{P}_n$ with $p \succ s$, p must overlap with z^*.[12] So all streaks in $\mathcal{P}_n$ that dominate s cannot be an analogous streak of z^*. Moreover, no streak in $\mathcal{N}_n$ dominates s, as s is dominated only by streaks in $\mathcal{P}_n$. The above statements imply that s is not dominated by any analogous streak of z^*, so that it is in the skyline of $\mathcal{AS}(z^*)$, and thus a historic moment of z^*. In summary, each streak in $\mathcal{U}_n$ is a historic moment of some streak in $\mathcal{SS}_n$ or $\mathcal{SS}_m$, and therefore needs to be kept. □

Combining the above lemmas, Theorem 4.19 follows.

Example 4.22. Consider Figure 2 when $\sigma = 0.75$; as in Example 4.18, we have:

- $\mathcal{P}_{19} = \{s_1, s_2, s_3, s_4, s_5\}$;
- $skyline(\mathcal{N}_{19}) = \{s_6, s_7, s_8, s_{11}\}$;
- in $skyline(\mathcal{N}_{19}) \cup \mathcal{P}_{19}$, s_8 is universally dominated by s_5; and
- each streak in $skyline(\mathcal{N}_{19}) \cup \mathcal{P}_{19}$ has length at least σ.

According to Theorem 4.19, minimal subset $\mathcal{U}_{19}$ of $\mathcal{LPS}_{19}$ is the set of streaks in $skyline(\mathcal{N}_{19}) \cup \mathcal{P}_{19}$ that (1) are not universally dominated by any streak in $\mathcal{P}_{19}$, which gives $s_8 \notin \mathcal{U}_{19}$, and (2) have length at least σ. So we have:

- $\mathcal{U}_{19} = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_{11}\}$ and
- $\mathcal{SS}_{19} = \{s_1, s_2, s_3, s_4\}$

4.3.1 SOIA Maintenance. Theorem 4.19 gives how to obtain $\mathcal{U}_n$ of data sequence D_n. Based on that, now we discuss how to obtain $\mathcal{U}_m$ accordingly when new values $\langle v_{n+1}, v_{n+2}, \ldots, v_m \rangle$ are appended to the data sequence D_n. Without loss of generality, we first focus on when a single data value v_{n+1} is appended, and then generalize the procedure when values $\langle v_{n+1}, v_{n+2}, \ldots, v_m \rangle$ are appended in a batch.

When appending value v_{n+1} to D_n, it works like BIA (Section 4.2.1) to get $\mathcal{SS}_{n+1}$ and local prominent streaks ending at n based on three different cases. Specifically, our maintenance algorithm aims to do the following:

[11]The reason is as follows: clearly, s is an analogous streak of z^*. Next, if to the contrary s is not in the skyline of $\mathcal{AS}(z^*)$, then there is some streak $y \in \mathcal{AS}(z^*)$ that dominates s. Such a y must not overlap with z^* (since it is an analogous streak of z^*) and must not include the value ϵ that is newly appended to D (since the value of y is at least $s.v$ for $y \succ s$). In other words, y must be a streak in $\mathcal{LPS}_n$. A contradiction occurs, for no streak in $\mathcal{LPS}_n$ should dominate s.

[12]Else, $\mathcal{SS}_n^{(p)}$ does not contain any situational streak of which s is an analogous streak, so that s is universally dominated by p, and therefore $s \notin \mathcal{U}_n$. A contradiction occurs.

M1. Identify streaks in case (1). Those streaks will not be in $\mathcal{SS}_{n+1}$ and they require no more isolation. So we insert them into the R-tree.
M2. Compute the updated set of situational streaks $\mathcal{SS}_{n+1}$ from case (2) and case (3). The streaks in $\mathcal{SS}_{n+1}$ will be stored explicitly, sorted by their values.
M3. Resume minimality, as minimality may be violated at this point.

The reasons causing the minimality violation include the following:

(a) A perplexing streak $s \in \mathcal{P}_n$ may become a nonperplexing streak since D_{n+1}. For D_n, s can only prune streaks that it universally dominates through Lemma 4.16. But suppose that s further becomes a nonperplexing streak for D_{n+1}. Then, s is in $\mathcal{N}_{n+1}$ and can prune any streak that it dominates through Lemma 4.12. Recall that universally domination is harder to achieve than ordinary domination. When s goes from $\mathcal{P}_n$ to $\mathcal{N}_{n+1}$, it means now its "pruning power" has increased. So now more streaks could be pruned by s and they should be removed accordingly in order to maintain minimality.
(b) A streak s, which is not universally dominated by any perplexing streak in $\mathcal{P}_n$, may now be *universally dominated* by a perplexing streak in $\mathcal{P}_{n+1}$, so that s should be removed. This happens when s is an analogous streak of some streak z in $\mathcal{SS}_n$ but z is no longer in $\mathcal{SS}_{n+1}$ due to case (1), or s is no longer the analogous streak of the extended z' in $\mathcal{SS}_{n+1}$ because of case (2) (i.e., $|z'|$ becomes longer and $|s| \geq |z'| \cdot \sigma$ does not hold anymore). Note that when s is not an analogous streak of any streak in $\mathcal{SS}_{n+1}$ and is dominated by some streak in $\mathcal{P}_{n+1}$, s could be and should be removed in order to maintain minimality.

Algorithm 4 shows how the maintenance procedure works in the case of appending a single value.

ALGORITHM 4: SOIA Maintenance Procedure (Only for Appending a Single Value)

```
1:  procedure MAINTENANCE($v_{n+1}$)
2:      if $n == 0$ then
3:          $\mathcal{P}_1 = \{(1, 1, v_1)\}$;
4:          return;
5:      $\mathcal{SS}_{n+1} = \varnothing$;
6:      for each streak $(i, n, v)$ in $\mathcal{SS}_n$ do
7:          if $v_{n+1} \geq v$ then
8:              Insert $(i, n+1, v)$ to $\mathcal{SS}_{n+1}$;
9:          else if $|n - i| \geq \sigma$ then
10:             Insert $(i, n, v)$ to R-tree;
11:     if no streak in $\mathcal{SS}_n$ has value $v_{n+1}$ then
12:         if all streaks in $\mathcal{SS}_n$ have value $< v_{n+1}$ then
13:             Insert $(n+1, n+1, v_{n+1})$ to $\mathcal{SS}_{n+1}$;
14:         else
15:             Select the streak $(i, n, v)$ in $\mathcal{SS}_n$ whose value $v > v_{n+1}$ and is the smallest;
16:             Extend it to be $(i, n+1, v_{n+1})$
17:             Insert it into $\mathcal{SS}_{n+1}$;
18:     $\mathcal{T}$ = streaks in R-tree;
19:     Set $\mathcal{P}_n^{\mathcal{T}} = \mathcal{T} \cap \mathcal{P}_n$;
20:     Find $\Delta\mathcal{N}$ and $\mathcal{P}_{n+1}$ from $\mathcal{P}_n^{\mathcal{T}}$ and $\mathcal{SS}_{n+1}$;
21:     for each streak $y$ in $\Delta\mathcal{N}$ do
22:         Remove streak $s$ from R-tree if $y \succ s$;
23:     for each streak $p$ in $\mathcal{P}_{n+1}$ do
24:         Remove streak $s$ from R-tree if $s$ is universally dominated by $p$;
```

Lines 2 to 4 are to initialize $\mathcal{P}_1$. Lines 6 to 17 work like BIA's maintenance, except that it needs to check the length of streak when inserting to the R-tree (Line 9). Lines 18 to 24 are for resuming the minimality, corresponding to M3(a) and M3(b) discussed above. Specifically, we are meant to do two kinds of pruning:

- Pruning (i): Remove the streaks that are dominated by any streak in $skyline(\mathcal{N}_{n+1})$ (c.f. Lemma 4.12).
- Pruning (ii): Remove the streaks that are universally dominated by any streak in $\mathcal{P}_{n+1}$ (c.f. Lemma 4.16).

Let $\mathcal{T}$ be the streaks in the updated R-tree after executing Lines 2 to 14 of the maintenance procedure. So to carry out Pruning (i) and (ii) above, a basic method is to identify $\mathcal{N}_{n+1}$ by comparing each streak in $\mathcal{T}$ with the streaks in $\mathcal{SS}_{n+1}$, according to Definition 4.10. Once $\mathcal{N}_{n+1}$ is obtained, Pruning (i) above can be done. Next, identify streaks in $\{\mathcal{T} - N_{n+1}\} \cup \mathcal{SS}_{n+1}$ as $\mathcal{P}_{n+1}$, and use it to complete Pruning (ii).

For efficiency, we now show that Pruning (i) can be done by a much smaller subset $\Delta\mathcal{N} \subseteq \mathcal{N}_{n+1}$ instead, where $\Delta\mathcal{N} = \mathcal{N}_{n+1} - \mathcal{N}_n$. Especially, $\Delta\mathcal{N}$ can be obtained from $\mathcal{P}_n^{\mathcal{T}}$, where $\mathcal{P}_n^{\mathcal{T}}$ denotes $\mathcal{P}_n \cap \mathcal{T}$. Also, $\mathcal{P}_{n+1}$ can be identified in a more efficient way accordingly.

Lemma 4.23. *If a streak in $\mathcal{T}$ is dominated by any streak in $skyline(\mathcal{N}_{n+1})$, then such a streak is dominated by some streak in $\Delta\mathcal{N}$.*

Proof. We prove the lemma with the help of the following two lemmas,

Lemma 4.24. *For a streak $s \in \mathcal{T}$, if $s \notin \mathcal{P}_n^{\mathcal{T}}$, then $s \notin \mathcal{P}_{n+1}$.*

Proof. Let s be a streak in the $\mathcal{T}$ but not in $\mathcal{P}_n$, i.e., $s \notin \mathcal{P}_n^{\mathcal{T}}$. Now, suppose to the contrary: $s \in \mathcal{P}_{n+1}$. This implies that there exists streak $z' = (i', j', v') \in \mathcal{SS}_{n+1}$, i.e., $j' = n + 1$, such that s overlaps with z', and $|s| \geq |z'| \cdot \sigma$. However, for $z' \in \mathcal{SS}_{n+1}$, there must be a situational streak $z = (i, j, v) \in \mathcal{SS}_n$ with the same starting position as z', i.e., $i = i'$ and $j = n$. It follows that s also overlaps with z, and $|s| \geq |z| \cdot \sigma$. Thus, $s \in \mathcal{P}_n^{\mathcal{T}}$, a contradiction. □

Lemma 4.25.

$$skyline(\mathcal{N}_{n+1}) = skyline(skyline(\mathcal{N}_n) \cup \Delta\mathcal{N}).$$

Proof. By Lemma 4.24, for any streak $s \in \mathcal{N}_n$, which means $s \notin \mathcal{P}_n^{\mathcal{T}}$, $s \notin \mathcal{P}_{n+1}$. Thus, by definition, such a streak s must be in $\mathcal{N}_{n+1}$. This implies $\mathcal{N}_n \subseteq \mathcal{N}_{n+1}$. So we have $\mathcal{N}_{n+1} = \mathcal{N}_n \cup \Delta\mathcal{N}$. Further, if a streak s is in $\mathcal{N}_n$ but not in $skyline(\mathcal{N}_n)$, it is not in $skyline(\mathcal{N}_{n+1})$, since the streak that dominates s is still in $\mathcal{N}_{n+1}$. That completes the proof since removing a nonskyline streak from a set would not affect the result of a skyline operation. □

Lemma 4.25 implies that if a streak in $\mathcal{T}$ is dominated by $skyline(\mathcal{N}_{n+1})$, then it is dominated by $skyline(\mathcal{N}_n) \cup \Delta\mathcal{N}$. Furthermore, if a streak is dominated by $skyline(\mathcal{N}_n)$, it does not exist in $\mathcal{T}$ because (1) $\mathcal{T}$ includes the streaks in $\mathcal{U}_n - \mathcal{SS}_n$ and the newly streaks ended at n by Line 8 of Algorithm 4, and none of these newly added streaks are dominated by $skyline(\mathcal{N}_n)$, and (2) all streaks in $\mathcal{T}$ that are dominated by $skyline(\mathcal{N}_n)$ are pruned during any previous execution of the maintenance procedure. Thus, any streak in $\mathcal{T}$ that is dominated by $skyline(\mathcal{N}_{n+1})$ must be dominated by some streak in $\Delta\mathcal{N}$. □

We proceed to discuss how to obtain $\Delta\mathcal{N}$ and $\mathcal{P}_{n+1}$. First, we state that $\Delta\mathcal{N}$ can be obtained from $\mathcal{P}_n^{\mathcal{T}}$.

Lemma 4.26. $\Delta\mathcal{N} \subseteq \mathcal{P}_n^{\mathcal{T}}$.

Proof. Any streak $s \in \mathcal{T}$ that s in $\Delta\mathcal{N} \subseteq \mathcal{N}_{n+1}$ cannot be in $\mathcal{P}_{n+1}$, so that its ending position is at most n. This implies $s \in \mathcal{N}_n \cup \mathcal{P}_n^{\mathcal{T}}$. By definition, $\Delta\mathcal{N} = \mathcal{N}_{n+1} - \mathcal{N}_n$. So s cannot be in $\mathcal{N}_n$ but can only be in $\mathcal{P}_n^{\mathcal{T}}$, which gives $\Delta\mathcal{N} \subseteq \mathcal{P}_n^{\mathcal{T}}$. □

Lemma 4.27. $\Delta\mathcal{N} = \mathcal{N}_{n+1} \cap \mathcal{P}_n^{\mathcal{T}}$.

Proof. Following Lemma 4.26, we have $\Delta\mathcal{N} \subseteq \mathcal{P}_n^{\mathcal{T}}$ and then

- $\mathcal{N}_n \cap \mathcal{P}_n^{\mathcal{T}} = \mathcal{N}_n \cap (\mathcal{P}_n \cap \mathcal{T}) = (\mathcal{N}_n \cap \mathcal{P}_n) \cap \mathcal{T} = \varnothing$, and
- $\mathcal{N}_{n+1} \cap \mathcal{P}_n^{\mathcal{T}} = (\mathcal{N}_n \cup \Delta\mathcal{N}) \cap \mathcal{P}_n^{\mathcal{T}} = (\mathcal{N}_n \cap \mathcal{P}_n^{\mathcal{T}}) \cup (\Delta\mathcal{N} \cap \mathcal{P}_n^{\mathcal{T}}) = \varnothing \cup \Delta\mathcal{N} = \Delta\mathcal{N}$.

The lemma follows. □

Next, we state that $\mathcal{P}_{n+1}$ can be obtained from $\mathcal{P}_n^{\mathcal{T}}$, $\Delta\mathcal{N}$, and the updated $\mathcal{SS}_{n+1}$:

Lemma 4.28. $\mathcal{P}_{n+1} = (\mathcal{P}_n^{\mathcal{T}} - \Delta\mathcal{N}) \cup \mathcal{SS}_{n+1}$.

Proof. We discuss $\mathcal{P}_{n+1}$ as two parts below:

(i) $\mathcal{P}_{n+1}$ includes $\mathcal{SS}_{n+1}$ by definition.
(ii) For streaks in $\mathcal{P}_{n+1} - \mathcal{SS}_{n+1}$, they are in $\mathcal{N}_n \cup \mathcal{P}_n^{\mathcal{T}}$. By Lemma 4.24, they must be in $\mathcal{P}_n^{\mathcal{T}}$; moreover, they cannot belong to $\mathcal{N}_{n+1}$. That is, they should be in $\mathcal{P}_n^{\mathcal{T}} - \mathcal{P}_n^{\mathcal{T}} \cap \mathcal{N}_{n+1}$, where $\mathcal{P}_n^{\mathcal{T}} \cap \mathcal{N}_{n+1} = \Delta\mathcal{N}$ by Lemma 4.27.

(i) gives $\mathcal{SS}_{n+1}$ and (ii) gives $\mathcal{P}_n^{\mathcal{T}} - \Delta\mathcal{N}$. The lemma thus follows. □

With Lemmas 4.26, 4.27, and 4.28, we shall obtain $\Delta\mathcal{N}$ and $\mathcal{P}_{n+1}$ as follows:

(1) For each streak p in $\mathcal{P}_n^{\mathcal{T}}$, compare p with all streaks in $\mathcal{SS}_{n+1}$.
   If there does not exist any streak z in $\mathcal{SS}_{n+1}$ such that p overlaps with z and $|p| \geq |z| \cdot \sigma$, then p belongs to $\mathcal{N}_{n+1}$. For efficiency's sake, we insert p to $\Delta\mathcal{N}$ (c.f. Lemma 4.23). Else, p belongs to $\mathcal{P}_{n+1}$ and we insert it into $\mathcal{P}_{n+1}$.
(2) Expand $\mathcal{P}_{n+1}$ by including all streaks in $\mathcal{SS}_{n+1}$.

The above gives the details of Line 20 in Algorithm 4. The remaining Lines 21 to 24 carry out the pruning based on the discussed lemmas so far.

Example 4.29. Let us revisit the example of appending $v_{20} = 5$ as in Figure 3, with $\sigma = 0.75$. Recall that before the value is appended as in Figure 2, we have:

- $\mathcal{SS}_{19} = \{s_1, s_2, s_3, s_4\}$,
- $\mathcal{P}_{19} = \{s_1, s_2, s_3, s_4, s_5\}$,
- $\mathcal{N}_{19} = \{s_6, s_7, s_8, s_9, s_{10}, s_{11}\}$, and
- Rtree $= \{s_5, s_6, s_7, s_{11}\}$ because s_9 and s_{10} are not in $skyline(\mathcal{N}_{19})$ and s_8 is universally dominated by s_5.

Appending v_{20} requires us to compute the new $\mathcal{SS}_{20}$ and local prominent streaks ended at 19. Running Lines 6 to 18 will result in:

- $\mathcal{SS}_{20} = \{s_3'(8, 20, 4), s_4'(1, 20, 1), s_{new}(14, 20, 5)\}$, as explained in Example 4.2, and
- $\mathcal{T} = \{s_1, s_2, s_5, s_6, s_7, s_{11}\}$, as s_1 and s_2 get inserted into the R-tree according to case M1 (Line 10 in Algorithm 4).

Lines 19 and 20 then identify the following:

- $\mathcal{P}_{19}^{\mathcal{T}} = \mathcal{T} \cap \mathcal{P}_{19} = \{s_1, s_2, s_5\}$.
- $\Delta\mathcal{N} = \{s_5\}$ because in $\mathcal{P}_{19}^{\mathcal{T}}$, only s_5 has length less than $|z| \cdot \sigma$, where z denotes any streak in $\mathcal{SS}_{20}$.
- $\mathcal{P}_{20} = (\mathcal{P}_{19}^{\mathcal{T}} - \Delta\mathcal{N}) \cup \mathcal{SS}_{20} = \{s_1, s_2, s_{new}, s_3', s_4'\}$.

Lines 21 and 22 remove s_7 from the R-tree because $s_5 \in \Delta\mathcal{N}$ and $s_5 \succ s_7$. Next, Lines 23 and 24 check if any streak in the R-tree is universally dominated by streaks in $\mathcal{P}_{20}$ and prune them accordingly. In the example, no such streaks are found, and the minimality of the streaks stored in R-tree is maintained.

Now we generalize the discussions above to handle the case that a batch of data values $\langle v_{n+1}, v_{n+2}, \ldots, v_m \rangle$ is appended to D_n that results in D_m. Algorithm 5 shows the pseudo-code of this maintenance procedure, and we have the following:

- Upon each value $v_{n+k} \in \{v_{n+1}, v_{n+2}, \ldots, v_m\}$, Lines 3 to 23 work similarly to Lines 2 to 20 in Algorithm 4 that it computes corresponding to $\mathcal{P}_{n+k}$ and $\Delta\mathcal{N}$, respectively. $\mathscr{N}$ is used to collect the streaks in $\Delta\mathcal{N}$ generated in Line 22 for each value v_{n+k}, so that in the end we have $\mathscr{N} = \mathcal{N}_m - \mathcal{N}_n$.

ALGORITHM 5: SOIA Maintenance Procedure

```
1:  procedure MAINTENANCE($\langle v_{n+1}, v_{n+2}, \ldots, v_m \rangle$)
2:      $\mathscr{N} = \varnothing$;
3:      for $k = 1$ to $m - n$ do
4:          if $n == 0$ and $k == 1$ then
5:              $\mathcal{P}_1 = \{(1, 1, v_1)\}$;
6:              continue;
7:          $\mathcal{SS}_{n+k} = \varnothing$;
8:          for each streak $(i, n + k - 1, v)$ in $\mathcal{SS}_{n+k-1}$ do
9:              if $v_{n+k} \geq v$ then
10:                 Insert $(i, n + k, v)$ to $\mathcal{SS}_{n+k}$;
11:             else if $|n + k - 1 - i| \geq \sigma$ then
12:                 Insert $(i, n + k - 1, v)$ to Buffer $B$;
13:         if no streak in $\mathcal{SS}_{n+k-1}$ has value $v_{n+k}$ then
14:             if all streaks in $\mathcal{SS}_{n+k-1}$ have value $< v_{n+k}$ then
15:                 Insert $(n + k, n + k, v_{n+k})$ to $\mathcal{SS}_{n+k}$;
16:             else
17:                 Select the streak $(i, n + k - 1, v)$ in $\mathcal{SS}_{n+k-1}$ whose value $v > v_{n+k}$ and is the smallest;
18:                 Extend it to be $(i, n + k, v_{n+k})$
19:                 Insert it into $\mathcal{SS}_{n+k}$;
20:         $\mathcal{T}$ = streaks in R-tree $\cup$ $B$;
21:         Set $\mathcal{P}_{n+k-1}^{\mathcal{T}} = \mathcal{T} \cap \mathcal{P}_{n+k-1}$;
22:         Find $\Delta\mathcal{N}$ and $\mathcal{P}_{n+k}$ from $\mathcal{P}_{n+k-1}^{\mathcal{T}}$ and $\mathcal{SS}_{n+k}$;
23:         $\mathscr{N} = \mathscr{N} \cup \Delta\mathcal{N}$;
24:     Insert streaks in $B$ into R-tree;
25:     for each streak $y$ in $\mathscr{N}$ do
26:         Remove streak $s$ from R-tree if $y \succ s$;
27:     for each streak $p$ in $\mathcal{P}_m$ do
28:         Remove streak $s$ from R-tree if $s$ is universally dominated by $p$;
```

- Similar to Algorithm 2, we store the new local prominent streaks in a Buffer B (Line 12) and then insert them into the R-tree in a batch (Line 24) to improve the I/O efficiency.
- Lines 25 to 28 are to prune the R-tree to guarantee its space-optimal properly, according to Lemma 4.23 and Lemma 4.16.

4.3.2 SOIA Lookup. The Lookup procedure of SOIA is the same as BIA's Lookup procedure. Algorithm 6 shows the pseudo-code when dealing with a new parameter σ' (with $\sigma' \geq \sigma$) and a new parameter k' from a user. It first obtains the top k' situational streaks by a linear scan of $\mathcal{SS}_n$, since streaks in $\mathcal{SS}_n$ are sorted by values. Then, for each such situational streak z, it constructs a query using the following region:

$$Q = [|z| \cdot \sigma', +\infty] \times [0, z.i) \times [z.v \cdot \sigma', +\infty].$$

That region contains exactly the set $\mathcal{AS}(z)$ of analogous streaks of z. We can thus apply the BBS skyline algorithm in [13] to compute $\mathcal{HM}(z)$ (the constrained skyline) from the R-tree.

ALGORITHM 6: SOIA Lookup Procedure

```
1: procedure Lookup($\sigma', k'$)
2:     $\mathcal{Z}$ = Get-Top-SS($\mathcal{SS}_n, k'$);
3:     for each streak $z$ in $\mathcal{Z}$ do
4:         $Q = [|z| \cdot \sigma', +\infty] \times [0, z.i) \times [z.v \cdot \sigma', +\infty]$;
5:         $\mathcal{HM}(z)$ = BBS(R-tree, $Q$); // compute constrained skyline
```

Example 4.30. Following Example 4.29, given $\sigma = 0.6$ and $k = 2$, now we have

- Rtree = $\{s_1, s_2, s_5, s_6, s_{11}\}$ and
- $\mathcal{SS}_{19} = \{s'_3, s'_4, s_{new}\}$.

When querying with parameter $\sigma' = 0.6$ and $k' = 2$: for s_{new}, find out its analogous streaks in R-tree that $\mathcal{AS}(s_{new}) = \{s_{11}\}$, and its historic moment is $skyline(\mathcal{AS}(s_{new})) = \{s_{11}\}$; for s'_3, there is no analogous streak for it. So now the historic moment is $\{s_{11}\}$.

5 CASE STUDY

5.1 Microsoft's Stock Price

The first sequence dataset is Microsoft's (NASDAQ:MSFT) daily stock price[13] from 1986 to 2014. We set the similarity threshold σ as 1 and monitor the historic moments of the top 1 situational streak. On June 11, 2014, we got the following top 1 situational streak:

(2014–6–05, 2014–6–11, 40.35),

and its corresponding historic moments are:

(1999–12–16, 2000–1–05, 40.36)
(1999–12–21, 2000–1–03, 41.52)
(1999–12–22, 2000–1–03, 41.77)
(1999–12–22,1999–12–31, 41.83)
(1999–12–22,1999–12–30, 42.08)
(1999–12–16, 2000–1–03, 40.40).

[13] http://finance.yahoo.com/q/hp?s=MSFT.

It turns out that all those historic moments *happened around the end of 1999*, meaning that Microsoft had not had such a long streak of high stock prices for almost 14 years. Indeed, a real news report[14] was given on June 11, 2014 based on the above:

> *Microsoft stock inching closer to all-time high. Don't look now, but Microsoft (NASDAQ:MSFT) stock is at a 14-year high and is approaching its all-time high reached just before the dot-com crash.*

5.2 Beijing's Temperature

The second sequence dataset is the average daily temperature of Beijing[15] from 1995 to 2014. We set the similarity threshold σ as 1 and monitor the historic moments of the top 1 situational streak. On July 29, 2010, we got the following top 1 situational streak:

$$(2010-7-27, 2010-7-29, 85.5),$$

and its corresponding historic moments are:

$$\begin{array}{l}(2000-7-03, 2010-7-06, 86.8)\\(2000-7-11, 2000-7-14, 87.6)\\(2000-7-22, 2000-7-26, 85.6)\\(1999-7-23, 1999-7-29, 87.4).\end{array}$$

These historic moments imply that the last time Beijing had such a long streak of high temperature was almost 10 years ago, and this observation was reported in the news on July 29, 2010:[16]

> *High temperature days in Beijing July last longest in the past ten years....*

5.3 Taiwan Seismic Datasets

The third dataset is the ground motion sensor stream of Taiwan.[17] Ground motion is the movement of the earth's surface. Seismologists can utilize ground motion data to study and even predict some geological activities such as earthquakes [2]. The datasets are streams of sample *counts*[18] for every 50ms, since 1998, with one data sequence per monitoring station. Each data sequence is about 350GB in size and BIA was required to build an index of size 500GB, exceeding the disk size of our experimental platform (detailed configuration in Section 6). In contrast, SOIA only required 2GB to house the index, making this case study feasible.

We set the similarity threshold σ as 0.9 and monitor the historic moments of the top 10 situational streaks. We got a number of interesting findings there when we studied the sequence data from one such station, namely, station "KMNB." Specifically, it illustrates that while historic moments that happened long ago are useful, those that happened very recently are also helpful to highlight the importance of a prominent streak in certain domains.

For example, we got a situational streak s_a on February 3, 2016. That was a streak of length of 202 units (i.e., 10.1s) with 40,334 counts. Let's look at its corresponding historic moments:

(2016-01-30-03:43, length=184, count=40391)
(2016-01-30-03:43, length=185, count=40188)
(2016-01-30-03:42, length=189, count=39161).

[14] http://www.scalper1.com/microsoft-stock-inching-closer-to-all-time-high.html.
[15] http://academic.udayton.edu/kissock/http/Weather/gsod95-current/CIBIEJNG.txt.
[16] http://news.163.com/10/0729/08/6COD18AV000146BC.html.
[17] http://ds.iris.edu/ds/nodes/dmc/data/
[18] *Count* is a scale unit of ground motion. A "count" value of 3.27508e9 indicates ground motion of 1 meter/second.

We can see that its historic moments just happened 4 days before February 3, 2016. That implied it was not a singular prominent event and was worth paying attention to. In fact, based on our record [16], an earthquake happened 2 days later, on February 5, 2016. As another example, we got another situational streak s_b on February 27, 2010, of length 238 units (i.e., 11.9s) with 42,492 counts. Its historic moments are:

(2010-02-26-01:49, length=219, count=40949)
(2010-02-26-01:48, length=223, count=40232)
(2010-02-26-01:48, length=230, count=40119).

It is important that its historic moments just happened 1 day before. Actually, an earthquake happened 5 days later, on March 4, 2010, according to [16].

How can one tell the above historic moments are fair indicators of earthquakes but not normal energy release? In order to answer that question, we ran SOIA on the dataset and obtained the following statistics:

Among all (situational) streaks of length > 200 and value > 40,000 (e.g., s_a and s_b above), their corresponding historic moments happened 23 days ago, on average.

One plausible message of the above is that the more recent historic moments are to the situational streaks (and farther away than the mean, 23 days), the more significant they are as an earthquake indicator. To cross-check that claim, we found that for all situational streaks of length > 200 and value > 40,000 in the dataset, whenever their corresponding historic moments happened before 23+ days, no earthquake was reported within weeks after those situational streaks happened.

6 PERFORMANCE STUDY

In this section, we evaluate the performance of SOIA using five real sequence datasets and compare with BIA. The experimental platform has a 2.8GHz Intel i5 CPU, 8GB RAM, and 256GB hard disk. Table 2 lists the information about the datasets, which are ordered based on their data sizes. Dataset D5 was too large that BIA would require index space larger than our disk; therefore, we only used a fraction of D5 in order to make BIA runnable in this section.

6.1 Overall Comparison

We first look at the overall performance of BIA and SOIA when the full dataset is available. We evaluate the performance of BIA and SOIA in terms of their (1) index building time (maintenance time), (2) query time (lookup time), and (3) index space.

We use $\sigma = 0.5$ in building the index structure (Maintenance procedure). We create five user lookup workloads: W_1, W_2, W_3, W_4, W_5, where a workload W_i mimics a user exploring the dataset by trying out $10 \cdot i$ different k and σ' historic moment lookups. Therefore, W_1 consists of 10 lookups (e.g., a casual user) and W_5 consists of 50 lookups (e.g., a serious journalist). When reporting the lookup time, we report the average running time of W_1 to W_5. That represents the total wait time for a user in one interactive session. In the experiment, each lookup query randomly chose a value between 1 and 10 for k and randomly chose a value between σ and 1 as σ'.

Figure 4 shows the results. Since SOIA has to invest some time to identify and prune redundant streaks in order to maintain a minimal space index for the latter use, its one-off maintenance time is higher than BIA (Figure 4(a)). Nevertheless, we see that such investment is worthwhile because the lookup time of SOIA is 9.58× (D1) to 184.14× (D4) better than BIA (Figure 4(b)), which gives users a much shorter waiting time during the exploration. Figure 4(c) shows that the space requirement of SOIA is much smaller than that of BIA. On D5 (a largely trimmed version of Taiwan's ground

Table 2. Summary of Datasets

Dataset	Size (MB)	Length	Description
D1	14.1	1,074,637	household global minute-averaged current intensity from December 2006 to December 2008[19]
D2	21.3	1,600,237	household global minute-averaged active power from December 2006 to December 2009
D3	27.5	1,802,000	EEG time-series datasets[20]
D4	32.5	2,764,800	number of requests to World Cup 98 website per second from April to May 1998[21]
D5	52.4	3,456,000	Taiwan KMNB station ground motion from Jan. 1 2017, to Jan. 2, 2017[22]

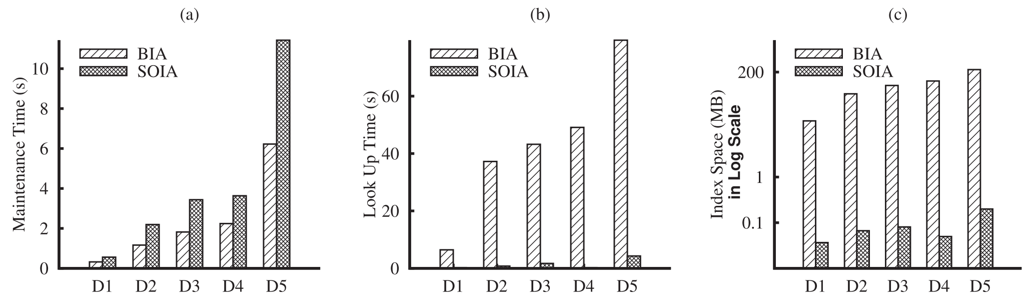


Fig. 4. BIA versus SOIA.

motion data), monitoring historic moments for just one station already requires BIA to build an index bigger than 200MB.

The minimal index size of SOIA is also the key factor that leads to its excellent historic moment lookup performance (other factors include the size and the value of the streak; see Algorithm 6). Compared with SOIA, BIA takes 656× (D1) to 2, 898× (D3) more index space. Table 3 lists the index space as well as the number of streaks stored using BIA and SOIA, respectively. For SOIA, the number of streaks in $Skyline(\mathcal{N}_n)$ and $\mathcal{P}_n$ is also reported. In fact, the sizes of the index structures are proportional to the numbers of streaks in $\mathcal{LPS}_n$. In the following sections, we only include the index size in our discussions.

6.2 Historic Moment Exploration with Data Update

Next, we look at the performance of SOIA and BIA for maintaining the index structure online. That is, whenever a value arrives, BIA inserts new local prominent streaks into the R-tree immediately, and SOIA maintains its space-optimal property by inserting and pruning the R-tree immediately.

In this experiment, we regard the first 98% of a dataset as the initial dataset and its index structure has been built by the MAINTENANCE procedure already. Then, we examine the performance of SOIA and BIA regarding a data append of the last $x\%$ of the dataset, where $x = 0.1, 0.5, 1$, and 2.

Figure 5 shows the experiment results. In the figures, we report:

(a) the time of the MAINTENANCE procedure in order to handle data appending,

[19] https://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption.
[20] http://alumni.cs.ucr.edu/~mueen/OnlineMotif/index.html.
[21] http://ita.ee.lbl.gov/html/contrib/WorldCup.html.
[22] http://ds.iris.edu/ds/nodes/dmc/forms/breqfast-request/.

Table 3. Number of Streaks Maintained by BIA and SOIA

Method	BIA		SOIA			
Dataset	Index Space (MB)	Number of Streaks Stored	Index Space (MB)	Number of Streaks Stored	Number of Streaks in $Skyline(\mathcal{N}_n)$	Number of Streaks in $\mathcal{P}_n$
D1	17.06	299,868	0.026	434	433	8
D2	66.91	1,175,934	0.061	1,012	987	31
D3	127.55	2,241,976	0.044	676	662	14
D4	101.87	1,790,601	0.080	1,362	1,337	25
D5	226.75	3,985,692	0.199	3,291	3,289	20

(b) the average time of running time of W_1 to W_5,
(c) the total of (a) and (b), and
(d) the size of the index after a data update.

The maintenance times of SOIA and BIA are similar (see subfigure (a)) because the advantage of having a smaller index in SOIA is offset by the extra effort spent on maintaining minimality. Nevertheless, SOIA is superior in terms of lookup performance (see subfigure (b)). So if one considers the time from getting the new data to the time that a user finishes a particular session of historic moment exploratory (see subfigure (c)), SOIA is much better than BIA. With no surprise, SOIA maintains a smaller index size all the way (see subfigure (d)).

6.3 Sensitivity Study

Here, we look at the impact of parameters σ and k on the performance of SOIA.

We first look at the influence of σ. In this experiment, given that the full dataset is available, we try different values for σ: 0.1, 0.3. 0.5. 0.7, 0.9. Figure 6(a) shows that different σ values do not influence the maintenance time of SOIA much. That is because the maintenance of SOIA is dominated by the time of scanning all $\mathcal{LPS}_n$ to get $\mathcal{P}_n$, which is independent of σ there. Figure 6(b) shows that a higher σ value during maintenance would make historic moment lookup more efficient because that would result in a smaller index as shown in Figure 6(c).

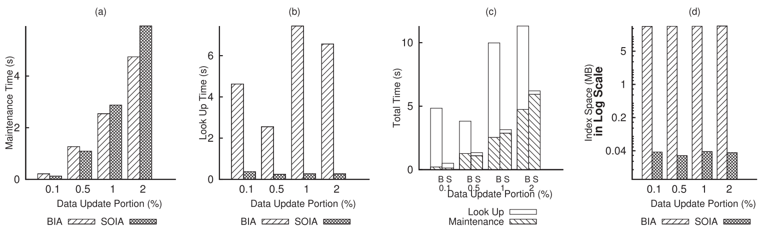
Figure 7 shows the time for maintaining space optimal online when data is being appended. In that experiment, we report the results of the maintenance time when the remaining 0.1%, 1%, and 2% of data are inserted, in case the index has been built for 98% of the original data already. When σ increases, the maintenance procedure takes less time. That is because a higher σ value would reduce the number of perplexing streaks, and for each perplexing streak p, the maintenance algorithm is required to remove streaks that are universally dominated by p from the R-tree (Algorithm 4, Lines 23 and 24). As a result, the maintenance time decreases when σ increases.

Lastly, we look at the impact of k on the performance of SOIA. Note that k has no impact on the maintenance phase of SOIA, so we only report the average execution time of the lookup phase based on the five workloads, with $\sigma = 0.5$.

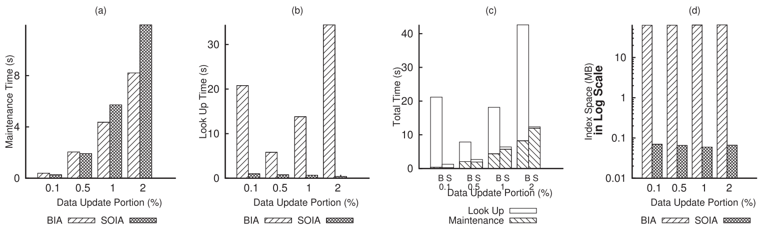
Figure 8 shows that the lookup time generally increases with the value k. That is because the lookup procedure looks for the historic moment for each top k situational streak. Increasing k would then increase the number of skyline computational calls to the index.

7 CONCLUSION

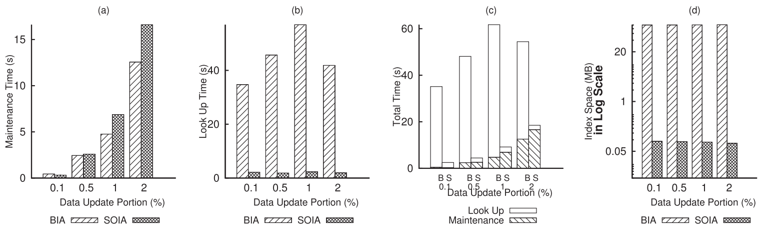
In this article, we introduce the notion of the historic moment, which complements existing work [8, 19] and provides holistic insights from sequence data. The computational issue of the



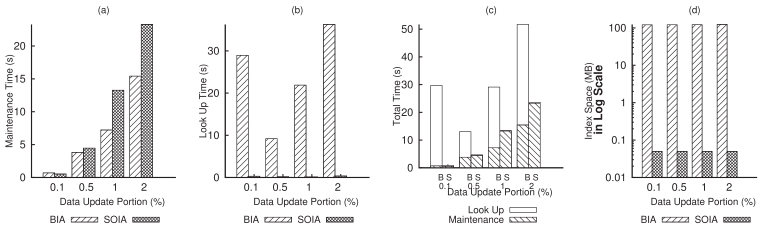
D1



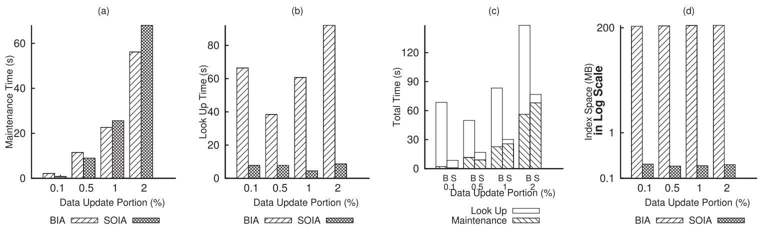
D2



D3



D4



D5

Fig. 5. BIA versus SOIA under data update.

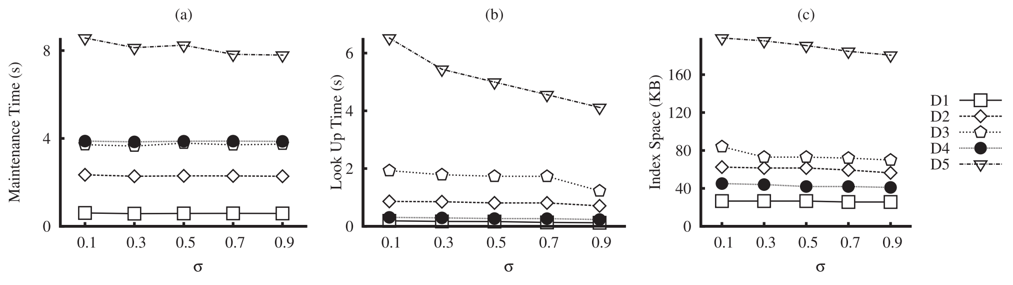


Fig. 6. Varying σ.

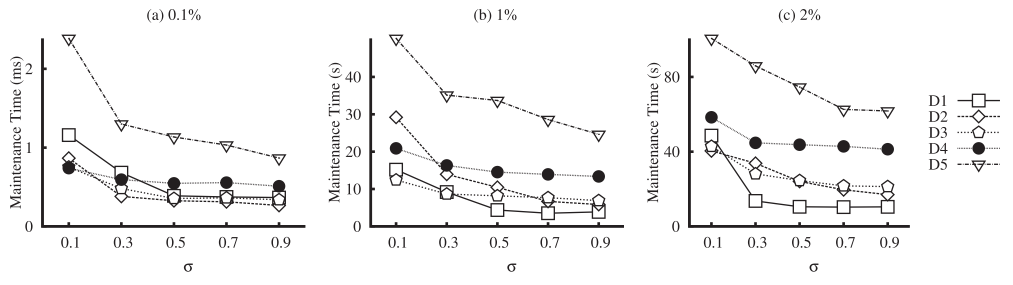


Fig. 7. Varying σ when updating.

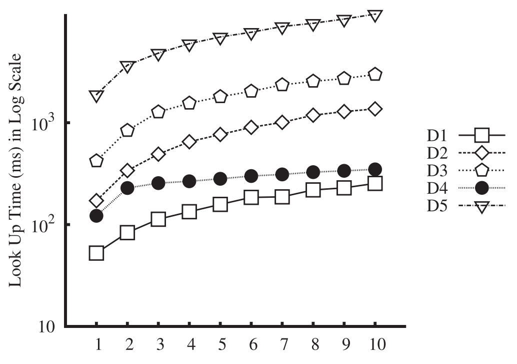


Fig. 8. Varying k.

historic moment focuses on the incremental and interactive aspects, in which new data are expected to arrive regularly and users are supposed to discover new historic moments right away, by feeding in different input parameters. To this end, we present SOIA, a highly efficient incremental algorithm using minimal space. Space optimality is important for online analysis and real-time monitoring systems because it significantly reduces the index size, thereby reducing the I/O per operation or making the index memory resident even when there are many data sequences. Case studies show that historic moments are helpful in computational journalism as well as in seismology. Experimental studies show that SOIA is both space and time efficient and outperforms the baseline on real data.

APPENDIX

A FINDING HISTORIC MOMENTS FROM MULTIPLE SEQUENCES

The main discussion focuses on finding historic moments in one data sequence. In this section, we also extend our problem and the algorithms to multiple data sequences (note that in [19], the authors also study how to find prominent streaks in multiple data sequences). We first give a motivating example from a piece of sports news in April 2014:[23]

> *"He (Kevin Durant) put up 25 points in his 40th consecutive game, which is the longest streak since Michael Jordan scored 25 in 40 consecutive games in the 1986-87 season. Wilt Chamberlain, who scored 25 or more in 80 consecutive games in 1960-61, holds the all time record in that category."*

In this piece of news, for the streak that Kevin Durant has achieved, the reporter reports historic moments from Michael Jordan and Wilt Chamberlain, which are two other data sequences.

Definition A.1 (Top k Situational Streak in Multiple Sequences). Given multiple sequences $\mathbb{D}_n = \{D_n^1, \ldots, D_n^w\}$ that contain w data sequences and each having n values, let $\mathcal{SS}_n^{D_n^1}, \ldots, \mathcal{SS}_n^{D_n^w}$ be the set of situational streaks for each data sequence, respectively. Let $\mathbb{SS}_n = \mathcal{SS}_n^{D_n^1} \cup \cdots \cup \mathcal{SS}_n^{D_n^w}$ be all situational streaks in $\mathbb{D}_n$. The top k situational streaks in $\mathbb{D}_n$ are the k streaks in $\mathbb{SS}_n$ with the highest values.

Definition A.2 (Analogous Streak in Multiple Sequences). Given multiple sequences $\mathbb{D}_n = \{D_n^1, \ldots, D_n^w\}$, let $\mathcal{LPS}_n^{D_n^1}, \ldots, \mathcal{LPS}_n^{D_n^w}$ be the local prominent streaks for each sequence, respectively. Let $\mathbb{LPS}_n = \mathcal{LPS}_n^{D_n^1} \cup \cdots \cup \mathcal{LPS}_n^{D_n^w}$ be all local prominent streaks set in $\mathbb{D}_n$. A local prominent streak $s \in \mathbb{LPS}_n$ is an *analogous streak* of a situational streak $z \in \mathbb{SS}_n$ when:

(1) $s.j < z.i$ (i.e., s ends before z starts),
(2) $|s| \geq |z| \cdot \sigma$ (i.e., the length of s is at least σ times that of z, where $\sigma \geq 0$ is a similarity threshold), and
(3) $s.v \geq z.v \cdot \sigma$ (i.e., the value of s is at least σ times that of z).

Definition A.3 (Historic Moments in Multiple Sequences). Given multiple sequences $\mathbb{D}_n = \{D_n^1, \ldots, D_n^w\}$, a similarity threshold σ, and a positive integer k. For each of situational streak z in the top k $\mathbb{SS}_n$, let $\mathbb{AS}(z)$ be the set of analogous streaks in multisequence $\mathbb{D}_n$ for z. Assume that each streak s in $\mathbb{AS}(z)$ is represented by a 4D point $(|s|, s.j, s.v, x)$, where x indicates which data sequence D^x that s comes from. The *historic moment* with respect to z, denoted by $\mathbb{HM}(z)$, is the skyline of $\mathbb{AS}(z)$ with respect to the **first three dimensions**, i.e., $|s|$, $s.j$, and $s.v$ of streak s.

In the sports news above, each data sequence in $\mathbb{D}_n$ represents the points that the player gets. On April 5, 2014, the streak with value 25 and length 40 from Kevin Durant is the top 1 situational streak. Its historic moments include the streak with value 25 and length 40 from Michael Jordan in 1986, as well as the streak with value 25 and length 80 from Wilt Chamberlain in 1960.

BIA and SOIA can be adapted to deal with this multisequence situation in a straightforward manner. We name them as BIA-MS and SOIA-MS, respectively.

A.1 BIA-MS

BIA-MS is largely similar to BIA in that $\mathbb{LPS}_n - \mathbb{SS}_n$ are inserted into the same R-tree, each $\mathcal{SS}_n^{D_n^i}$ is stored separately, and we have the following:

[23] http://ftw.usatoday.com/2014/04/kevin-durant-michael-jordan-record-thunder.

(1) Maintenance. Each sequence of new values $\langle v_{n+1}^{D^i}, v_{n+2}^{D^i} \ldots v_m^{D^i} \rangle$ for D^i calls the maintenance procedure of BIA, i.e., Algorithm 2, once.
(2) Lookup. Same as the lookup procedure of BIA, i.e., Algorithm 3.

A.2 SOIA-MS

SOIA-MS is largely similar to SOIA. SOIA-MS is also space optimal as it keeps minimal subset $\mathbb{U}_n$ of $\mathbb{LPS}_n$. It mainly generalizes the notion of *perplexing streaks* and *nonperplexing streaks* for multiple sequences:

Definition A.4 (Perplexing Streak). A streak $p \in \mathbb{LPS}_n$ is a *perplexing streak* when there exists a situational streak $z \in \mathbb{SS}_n$ such that:

(1) p overlaps with z,
(2) $|p| \geq |z| \cdot \sigma$, and
(3) $p.v \geq z.v \cdot \sigma$.

We use $\mathbb{P}_n$ to denote the set of all perplexing streaks in $\mathbb{D}_n$.

Definition A.5 (Nonperplexing Streak). Streaks in $\mathbb{LPS}_n$ that are not in $\mathbb{P}_n$ are *nonperplexing* streaks, denoted as $\mathbb{N}_n$. That is, $\mathbb{N}_n = \mathbb{LPS}_n - \mathbb{P}_n$. A streak $s \in \mathbb{LPS}_n$ is a nonperplexing streak if, for every situational streak $z \in \mathbb{SS}_n$ that s overlaps with, either $|s| < |z| \cdot \sigma$ or $s.v < z.v \cdot \sigma$.

Theorem A.6. *The minimal subset $\mathbb{U}_n$ of $\mathbb{LPS}_n$ is the set of streaks in $skyline(\mathbb{N}_n) \cup \mathbb{P}_n$ that (1) are not universally dominated by any streak in $\mathbb{P}_n$ and (2) have length at least σ.*

Proof. We prove the theorem by adapting the corresponding proof in Theorem 4.19. This is done by proving Lemma A.7 (the if case) and Lemma A.8 (the only-if case) below.

Lemma A.7. *If a streak s can serve as a historic moment of some streak in $\mathbb{SS}_n$ or $\mathbb{SS}_m$, then s is in $\mathbb{U}_n$.*

Proof. Firstly, $\mathbb{LPS}_n = \mathbb{N}_n \cup \mathbb{P}_n$ contains all local prominent streaks, and thus all historic moments for $\mathbb{D}_n$. Then, by adapting Lemma 4.12, we see that given that a streak is dominated by $skyline(\mathbb{N}_n)$, then it would not be a historic moment for either $\mathbb{SS}_n$ or $\mathbb{SS}_m$. So $skyline(\mathbb{N}_n) \cup \mathbb{P}_n$ contains all historic moments. Consequently, by adapting Corollary 4.17, we see that if a streak is universally dominated by some streak in $\mathbb{P}_n$, it cannot be the historic moment for $\mathbb{SS}_n$ or $\mathbb{SS}_m$. So $\mathbb{U}_n$ contains all historic moments. The lemma follows. □

Lemma A.8. *If a streak s is in $\mathbb{U}_n$, then s can serve as a historic moment of some streak in $\mathbb{SS}_n$ or $\mathbb{SS}_m$.*

Proof. Same as the proof of Lemma 4.21, with adapting D_n, $\mathcal{SS}_n$, $\mathcal{LPS}_n$, $\mathcal{P}_n$, and $\mathcal{AS}$ to D_n^i, $\mathbb{SS}_n$, $\mathbb{LPS}_n$, $\mathbb{P}_n$, and $\mathbb{AS}$, respectively. □

Combining the above lemmas, Theorem A.6 follows.

Algorithms 7 and 8 present the maintenance and lookup procedures of SOIA-MS, respectively. They are indeed a straightforward adaption of SOIA with minimal changes like computing $\mathcal{LPS}_n^{D_n^i}$ for every data sequence D_n^i and using the multiple sequence version of the definitions instead (e.g., using $\mathbb{P}_n$ instead of $\mathcal{P}_n$).

A.3 Performance Study

In this section, we compare the performance of BIA-MS and SOIA-MS using multiple real data sequences. The experimental platform and workload settings are the same as in Section 6. Among

ALGORITHM 7: SOIA-MS Maintenance

```
procedure Maintenance($\{\langle v^{D^1}_{n+1}, v^{D^1}_{n+2} \dots v^{D^1}_{m} \rangle, \dots, \langle v^{D^w}_{n+1}, v^{D^w}_{n+2} \dots v^{D^w}_{m} \rangle\}$)
    $\mathcal{N} = \varnothing$;
    for $k = 1$ to $m - n$ do
        if $n == 0$ and $k == 1$ then
            $\mathbb{P}_1 = \{(1, 1, v^{D^1}_1), (1, 1, v^{D^2}_1) \dots (1, 1, v^{D^w}_1)\}$;
            continue;
        $\mathbb{SS}_{n+k} = \varnothing$;
        for each new value $v^{D^j}_{n+k}$ do
            for each streak $(i, n+k-1, v)$ in $\mathcal{SS}^{D^j_{n+k-1}}_{n+k-1}$ do
                if $v^{D^j}_{n+k} \geq v$ then
                    Insert $(i, n+k, v)$ to $\mathcal{SS}^{D^j_{n+k}}_{n+k}$;
                else if $|n + k - 1 - i| \geq \sigma$ then
                    Insert $(i, n+k-1, v)$ to Buffer $B$, along with its data sequence information;
            if no streak in $\mathcal{SS}^{D^j_{n+k-1}}_{n+k-1}$ has value $v^{D^j}_{n+k}$ then
                if all streaks in $\mathcal{SS}^{D^j_{n+k-1}}_{n+k-1}$ have value $< v^{D^i}_{n+k}$ then
                    Insert $(n+k, n+k, v^{D^j}_{n+k})$ to $\mathcal{SS}^{D^j_{n+k}}_{n+k}$;
                else
                    Select the streak $(i, n+k-1, v)$ in $\mathcal{SS}^{D^j_{n+k-1}}_{n+k-1}$ with $v > v^{D^j}_{n+k}$ and is the smallest;
                    Extend it to be $(i, n+k, v^{D^j}_{n+k})$;
                    Insert it into $\mathcal{SS}^{D^j_{n+k}}_{n+k}$;
        $\mathbb{SS}_{n+k} = \mathcal{SS}^{D^1_{n+k}}_{n+k} \cup \dots \cup \mathcal{SS}^{D^w_{n+k}}_{n+k}$
        $\mathbb{T}$= streaks in R-tree $\cup B$;
        Set $\mathbb{P}^{\mathbb{T}}_{n+k-1} = \mathbb{T} \cap \mathbb{P}_{n+k-1}$;
        Find $\Delta\mathbb{N}$ and $\mathbb{P}_{n+k}$ from $\mathbb{P}^{\mathbb{T}}_{n+k-1}$ and $\mathbb{SS}_{n+k}$;
        $\mathcal{N} = \mathcal{N} \cup \Delta\mathbb{N}$;
    Insert streaks in $B$ into R-tree;
    for each streak $y$ in $\mathcal{N}$ do
        Remove streak $s$ from R-tree if $y > s$;
    for each streak $p$ in $\mathbb{P}_m$ do
        Remove streak $s$ from R-tree if $s$ is universally dominated by $p$;
```

ALGORITHM 8: SOIA-MS Lookup

```
1: procedure Lookup($\sigma', k'$)
2:     $\mathcal{Z}$ = Get-Top-SS($\mathbb{SS}_n, k'$);
3:     for each streak $z$ in $\mathcal{Z}$ do
4:         $Q = [|z| \cdot \sigma', +\infty] \times [0, z.i) \times [z.v \cdot \sigma', +\infty]$;
5:         $\mathbb{HM}(z)$ = BBS(R-tree, $Q$), and relevant data sequence information;
```

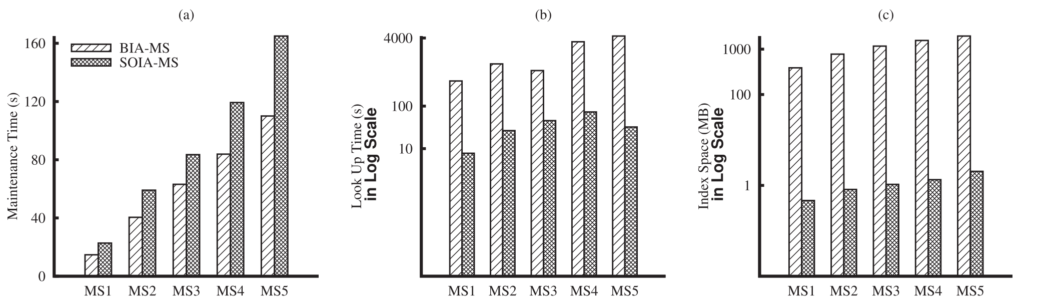


Fig. 9. SOIA-MS versus BIA-MS.

the five real datasets, only D5, the ground motion data sequence from seismic stations, has multiple sequences (from multiple stations). So we construct five real multisequence datasets as follows:

(1) MS1: KMNB, YHNB;
(2) MS2: KMNB, YHNB, YULB, TWGB;
(3) MS3: KMNB, YHNB, YULB, TWGB, TPUB, SSLB;
(4) MS4: KMNB, YHNB, YULB, TWGB, TPUB, SSLB, NACB, YOJ; and
(5) MS5: KMNB, YHNB, YULB, TWGB, TPUB, SSLB, NACB, YOJ, HK, HK0,

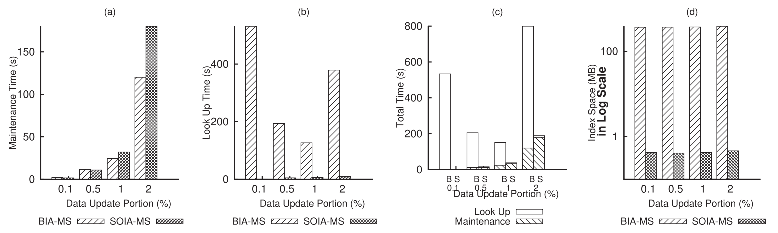
where each code (e.g., KMNB) above denotes one station name.

A.3.1 Overall Comparison. We first look at the overall performance of SOIA-MS and BIA-MS when the full datasets are available. Figure 9 shows the comparison results that we evaluate them in terms of their (1) index building time (MAINTENANCE time), (2) query time (LOOKUP time), and (3) index space.

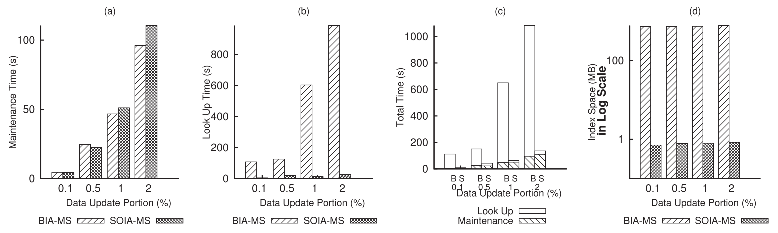
A.3.2 Historic Moment Exploration with Data Update. We evaluate the performance of SOIA-MS and BIA-MS for maintaining the index structure online. In this experiment, we regard the first 98% of a dataset as the initial dataset; i.e., each data sequence in the dataset has 98% as initial, and the index structure of it has been built by a maintenance procedure already.

Next, we examine the performance of SOIA-MS and BIA-MS regarding a data append of the last x% of each data sequence in the dataset, where $x = 0.1, 0.5, 1$, and 2. Figure 10 shows the experiment results. While SOIA-MS spends more effort in maintaining minimality, SOIA-MS is much more efficient when looking up the historic moments, as SOIA-MS always maintains a much smaller index size.

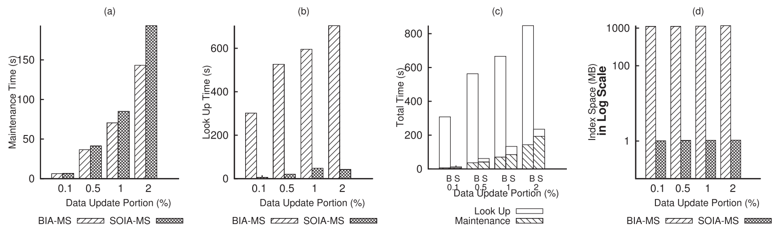
The basic principle of SOIA-MS is similar to SOIA, and the impact of parameters σ and k for SOIA-MS is also similar to SOIA. So here we don't repeat the experiments of parameter sensitivity.



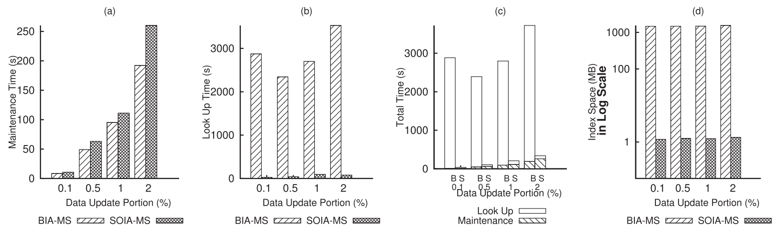
MS1



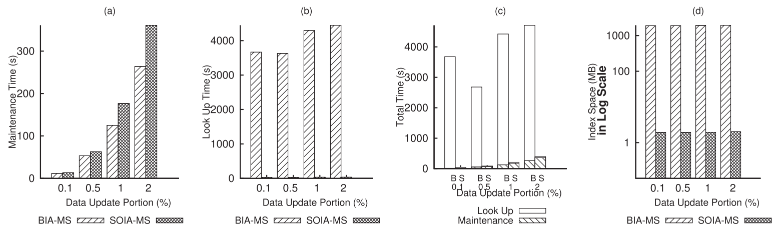
MS2



MS3



MS4



MS5

Fig. 10. SOIA-MS versus BIA-MS under data update.

REFERENCES

[1] Foto N. Afrati, Paraschos Koutris, Dan Suciu, and Jeffrey D. Ullman. 2015. Parallel skyline queries. *Theory of Computing Systems* 57, 4 (2015), 1008–1037.

[2] Gail M. Atkinson and David M. Boore. 2006. Earthquake ground-motion prediction equations for eastern North America. *Bulletin of the Seismological Society of America* 96, 6 (2006), 2181–2205.

[3] Wolf-Tilo Balke, Ulrich Güntzer, and Jason Xin Zheng. 2004. Efficient distributed skylining for web information systems. In *International Conference on Extending Database Technology*. Springer, 256–273.

[4] Stephan Borzsony, Donald Kossmann, and Konrad Stocker. 2001. The skyline operator. In *Proceedings of the 17th International Conference on Data Engineering*. IEEE, 421–430.

[5] Jan Chomicki, Parke Godfrey, Jarek Gryz, and Dongming Liang. 2005. Skyline with presorting: Theory and optimizations. In *Intelligent Information Processing and Web Mining*. Springer, 595–604.

[6] Sarah Cohen, James T. Hamilton, and Fred Turner. 2011. Computational journalism. *Communications of the ACM* 54, 10 (October 2011), 66–71. DOI: https://doi.org/10.1145/2001269.2001288

[7] Naeemul Hassan, Afroza Sultana, You Wu, Gensheng Zhang, Chengkai Li, Jun Yang, and Cong Yu. 2014. Data in, fact out: Automated monitoring of facts by FactWatcher. In *Proceedings of the 40th International Conference on Very Large Data Bases*. VLDB Endowment.

[8] Xiao Jiang, Chengkai Li, Ping Luo, Min Wang, and Yong Yu. 2011. Prominent streak discovery in sequence data. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1280–1288.

[9] Christos Kalyvas and Theodoros Tzouramanis. 2017. A survey of skyline query processing. *arXiv preprint arXiv:1704.01788* (2017).

[10] Donald Kossmann, Frank Ramsak, and Steffen Rost. 2002. Shooting stars in the sky: An online algorithm for skyline queries. In *Proceedings of the 28th International Conference on Very Large Data Bases*. VLDB Endowment, 275–286.

[11] Ming-Yen Lin, Chao-Wen Yang, and Sue-Chen Hsueh. 2016. Efficient computation of group skyline queries on MapReduce. *GSTF Journal on Computing (JoC)* 5, 1 (2016), 69–76.

[12] Dimitris Papadias, Yufei Tao, Greg Fu, and Bernhard Seeger. 2003. An optimal and progressive algorithm for skyline queries. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*. ACM, 467–478.

[13] Dimitris Papadias, Yufei Tao, Greg Fu, and Bernhard Seeger. 2005. Progressive skyline computation in database systems. *ACM Transactions on Database Systems* 30, 1 (March 2005), 41–82. DOI: https://doi.org/10.1145/1061318.1061320

[14] Afroza Sultana, Naeemul Hassan, Chengkai Li, Jun Yang, and Cong Yu. 2014. Incremental discovery of prominent situational facts. In *Proceedings of the IEEE 30th International Conference on Data Engineering*. IEEE.

[15] Kian-Lee Tan, Pin-Kwang Eng, and Beng Chin Ooi. 2001. Efficient Progressive Skyline Computation. In *Proceedings of the 27th International Conference on Very Large Data Bases (VLDB'01)*, Peter M. G. Apers, Paolo Atzeni, Stefano Ceri, Stefano Paraboschi, Kotagiri Ramamohanarao, and Richard Thomas Snodgrass (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, 301–310.

[16] Wikipedia. 2017. 2016 Taiwan earthquake — Wikipedia, The Free Encyclopedia. Retrieved from https://en.wikipedia.org/w/index.php?title=2016_Taiwan_earthquake&oldid=766638710 (accessed February 27, 2017).

[17] You Wu, Pankaj K. Agarwal, Chengkai Li, Jun Yang, and Cong Yu. 2012. On one of the few objects. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1487–1495.

[18] Yidong Yuan, Xuemin Lin, Qing Liu, Wei Wang, Jeffrey Xu Yu, and Qing Zhang. 2005. Efficient computation of the skyline cube. In *Proceedings of the 31st International Conference on Very Large Data Bases*. VLDB Endowment.

[19] Gensheng Zhang, Xiao Jiang, Ping Luo, Min Wang, and Chengkai Li. 2014. Discovering general prominent streaks in sequence data. *ACM Transactions on Knowledge Discovery from Data* 8, 2, Article 9 (June 2014), 37 pages. DOI: https://doi.org/10.1145/2601439

Received June 2017; revised August 2018; accepted September 2018