Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2010, Article ID 828035, 15 pages
doi:10.1155/2010/828035

Research Article

Effective Space Usage Estimation for Sliding-Window Skybands

Lijun Chen,[1] Jiakui Zhao,[1] Qun Huang,[1] and Liang Huai Yang[2]

[1] *School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China*
[2] *College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China*

Correspondence should be addressed to Lijun Chen, ljchen99@gmail.com

Received 1 February 2010; Accepted 24 March 2010

Academic Editor: Ming Li

Copyright © 2010 Lijun Chen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Skyline query computes all the "best" elements which are not dominated by any other elements and thus is very important for decision-making applications. Recently, it is generalized to *skyband* query and a k-skyband query returns those elements dominated by no more than k, of other elements. To incorporate the skyband operator into the stream engine for monitoring skybands over sliding windows, space usage estimation for skyband operator becomes a critical issue in the query optimizer. In this paper, we firstly introduce the skyband sketch as the cost model. Based on the cost model, we propose an approach for estimating the space usage of skyband operator over sliding windows of data streams under the assumptions of statistical independence across dimensions, no duplicate values over each dimension, and dimension domains totally ordered. Experiments verify that our approaches can estimate the space usage effectively over arbitrarily distributed data. To the best of our knowledge, this is the first work that attempts to address the issue and proposes effective approaches to solve it.

1. Introduction

Skyline queries [1] are very important for multicriteria decision-making applications, as the queries can return all the "best" elements which are not dominated by any other element. However, skyline queries may eliminate elements which are valuable but dominated by few other elements, for dimensions commonly can not cover all user's consideration. Therefore, Papadias et al. [2] generalized the skyline to skyband, and a k-skyband query returns all the elements which are dominated by no more than k of other elements.

By using the common hotel example in the literature, assuming that each hotel has the information of its distance from the beach and its price, and that one prefers the hotels which are cheap and close to the beach, Figure 1 demonstrates the difference between the skyline

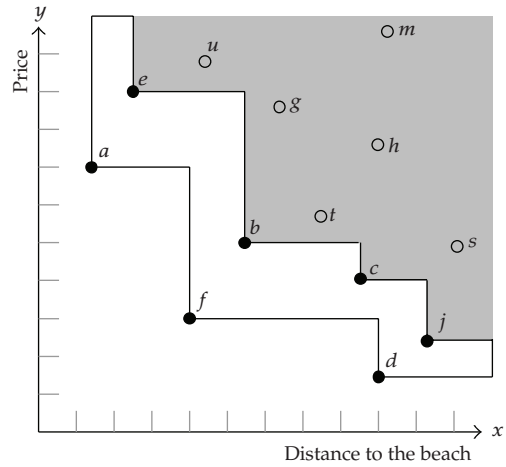


Figure 1: Skyline versus 1-skyband.

(the 0-skyband) and the 1-skyband. Three hotels, that is, a, d, and f, are returned by the skyline query, but additional four hotels, that is, b, c, e, and j, are returned by the 1-skyband query because they are dominated by only one of other elements. Buchta [3] proposed that the expected number of the skyline elements in a d-dimensional space which contains n elements is $\Theta((\ln n)^{d-1}/(d-1)!)$; therefore, low-dimensional skyline queries commonly return a small number of skyline elements to the user, and some valuable elements may be eliminated, the reason is that each element has a high probability of being dominated by other elements in a low-dimensional space. Skyband queries may return the elements which are valuable but dominated by few other elements to the user, hence, are widely used by decision-making applications in low-dimensional spaces.

Recently, the database research community witnessed a paradigm shift to continuous queries, and much attention has been put on sliding-window skyline queries [4, 5] in the stream environment. However, the issue of space usage estimation, which is very important for extending the query optimizer's cost model to accommodate skyline queries in the stream engine, is still left untouched. In this paper, we propose some effective approaches to estimate the space usage of sliding-window skyband queries. Since the skyline query is a special case of skyband queries, our proposed approaches can be naturally applied to sliding-window skyline queries as well.

Monitoring sliding-window skybands needs to extract all skyband elements from the live elements in the window and continuously report skyband changes as the window slides. In this paper, we first introduce the *skyband sketch* as the cost model and present effective policies for the sketch maintenance. As such, the skyband sketch has the quality of good space efficiency because it only stores the skyband elements along with the *potential-skyband elements* which do not belong to the skyband currently and are not guaranteed to be excluded from the skyband in their remaining lifespan. Next, under the assumption of statistical independence across dimensions, which is commonly used by query optimizers, and that no duplicate values exist over each dimension and domains are all totally ordered, we propose an approach for estimating the space usage of monitoring skybands over sliding windows. Experimental study verifies that our approaches can estimate the space usage effectively over arbitrarily distributed data. To the best of our knowledge, this is the first work that attempts to address the issue of space estimation and proposes effective approaches to solve it.

The rest of this paper is organized as follows. Section 2 summarizes the related work; Section 3 introduces some preliminary knowledge; Section 4 details our approaches for estimating the space usage; experimental results are given in Section 5 and followed by our conclusions in Section 6.

2. Related Work

Many algorithms have been proposed for computing static skylines, including the non-index-based algorithms [1, 6, 7] and the index-based algorithms [8–10], where the index-based algorithms uniformly outperform the non-index-based algorithms. Skyline computation under some certain conditions also received much attention, including skyline computation with partially ordered domains [11] and low-cardinality domains [12], subspace skyline computation [13, 14], skyline cube maintenance [15–19], and skyline computation in the distributed environment [14, 20–23]. Some skyline variations have also been proposed, including the k-dominant skyline [24], the top-k subspace skyline [25], the reverse skyline [26], the k most representative skyline [27], the probabilistic skyline [28], and the skyband [2].

Under the assumptions of statistical independence across dimensions, no duplicate values over each dimension, and dimension domains being all totally ordered, the problem of estimating the number of the skyline elements, that is, the skyline cardinality, has been addressed in the works [3, 29, 30]. Chaudhuri et al. [31] relaxed the assumption of no duplicate values over each dimension by allowing two possible values (e.g., 0 and 1).

As stated before, continuous skyline queries over sliding windows in data streams [4, 5] have important applications such as environment monitoring and trends sensing. To accommodate skyline operator in the stream processing engine, the issue of space usage estimation needs to be solved. Motivated by this ambition, under the similar assumptions, we propose robust approaches to estimate the number of the skyband and potential-skyband elements over continuously distributed data.

3. Preliminaries

In this section, we present some preliminary results that will be used in the next section. In addition, we also describe a data structure called *the skyband sketch*. Theorem 3.1 characterizes the number of the elements in a finite set which just satisfy k of the m properties. It is based on the generalized form of the *Inclusion-Exclusion Principle* [32]. Similarly, Theorem 3.2 characterizes the number of the elements in a finite set which satisfy no more than k of the m properties; the theorem will be used for our theoretical analysis of the space usage in the next section.

Theorem 3.1. *Suppose that $\mathcal{S}$ is a finite set, $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_m$ are m properties, and $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_m$ are m subsets of $\mathcal{S}$, where $\mathcal{S}_i$ consists of all those elements in $\mathcal{S}$ with property $\mathcal{P}_i$. Let $\Delta(m, k)$ be the number of the elements in $\mathcal{S}$ which just satisfy k of the m properties, it can be characterized as*

$$\Delta(m,k) = \sum_{i=k}^{m} (-1)^{i-k} \binom{i}{k} \tau(i), \tag{3.1}$$

where $\tau(i)$ $(0 \le i \le m)$ *is characterized as follows:*

$$\begin{aligned}
\tau(0) &= |\mathcal{S}|,\\
\tau(1) &= \sum_{i=1}^{m} |\mathcal{S}_i|,\\
\tau(2) &= \sum_{1\le i_1<i_2\le m} |\mathcal{S}_{i_1} \cap \mathcal{S}_{i_2}|,\\
\tau(3) &= \sum_{1\le i_1<i_2<i_3\le m} |\mathcal{S}_{i_1} \cap \mathcal{S}_{i_2} \cap \mathcal{S}_{i_3}|,\\
&\vdots\\
\tau(m) &= |\mathcal{S}_1 \cap \mathcal{S}_2 \cap \cdots \cap \mathcal{S}_m|.
\end{aligned} \tag{3.2}$$

Theorem 3.2. *Suppose that* $\mathcal{S}$ *is a finite set,* $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_m$ *are* m *properties, and* $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_m$ *are* m *subsets of* $\mathcal{S}$*, where* $\mathcal{S}_i$ *consists of all the elements in* $\mathcal{S}$ *which satisfy* $\mathcal{P}_i$*; the number of the elements in* $\mathcal{S}$ *which satisfy no more than* k *of the* m *properties, that is,* $\Gamma(m,k)$*, can be characterized as*

$$\Gamma(m,k) = \tau(0) + \sum_{i=k+1}^{m} (-1)^{i-k} \binom{i-1}{k} \tau(i), \tag{3.3}$$

where $\tau(i)$ $(0 \le i \le m)$ *is the same as that in Theorem 3.1.*

Proof. By Theorem 3.1, $\Gamma(m,k)$ can be characterized as

$$\begin{aligned}
\Gamma(m,k) &= \sum_{j=0}^{k} \Delta(m,j) = \sum_{j=0}^{k} \sum_{i=j}^{m} (-1)^{i-j} \binom{i}{j} \tau(i)\\
&= \sum_{i=0}^{k} \sum_{j=0}^{i} (-1)^{i-j} \binom{i}{j} \tau(i) + \sum_{i=k+1}^{m} \sum_{j=0}^{k} (-1)^{i-j} \binom{i}{j} \tau(i)\\
&= \tau(0) + \sum_{i=k+1}^{m} (-1)^{i-k} \binom{i-1}{k} \tau(i).
\end{aligned} \tag{3.4}$$

We have thus proved the theorem. □

In a d-dimensional space, for simplicity and without loss of generality, an element $\xi_\bullet$ is said to dominate another element $\xi_\circ$ if it is smaller than or equal to $\xi_\circ$ over each dimension and strictly smaller than $\xi_\circ$ over at least one dimension and is noted as $\xi_\bullet \succ \xi_\circ$. In a sliding-window, if no more than k of other live elements can dominate an element, the element is a k-skyband element; if an element is not a k-skyband element and no more than k of the succeeding elements can dominate it, the element is a potential-k-skyband element.

Now we are able to describe a data structure called *the skyband sketch* for keeping the k-skyband elements or the potential-k-skyband elements. The skyband sketch is a memory

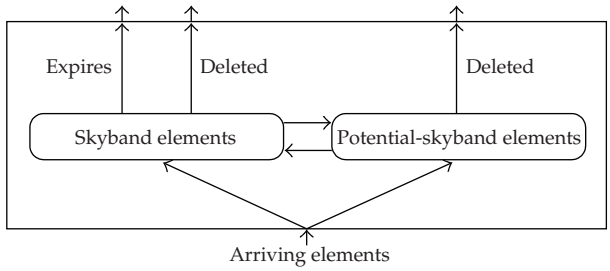


Figure 2: The architecture of the skyband sketch.

resident synopsis. The potential-skyband elements are the elements which do not belong to the skyband currently but are not guaranteed to be excluded from the skyband in their remaining lifespan. Hence the skyband sketch has the quality of good space efficiency for monitoring skybands over sliding-windows. The space usage in this paper is measured by the numbers of the skyband and the potential-skyband elements stored by the sketch.

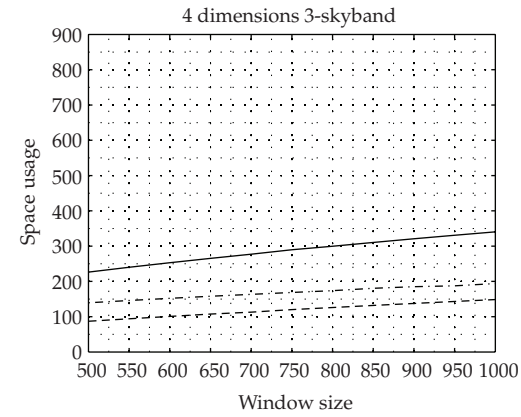
Figure 2 shows the architecture of the skyband sketch; the sketch changes occur only when a new element arrives or a current skyband element expires. When a new element arrives, if no more than k skyband elements can dominate it, it is probably a skyband element; otherwise, it is a potential-skyband element. If the new element appears to be a skyband element, all the skyband elements which are dominated by more than k succeeding skyband elements and all the potential-skyband elements which are dominated by more than k succeeding skyband and potential-skyband elements should be deleted because they will be dominated by the succeeding k elements during their remaining lifespan; in addition, the skyband elements which are dominated by no more than k succeeding skyband elements but are dominated by more than k live skyband elements will appear to be potential-skyband elements. If the new element appears to be a potential-skyband element, all potential-skyband elements which are dominated by more than k succeeding skyband and potential-skyband elements should be deleted. When a skyband element expires, all the potential-skyband elements which are dominated by no more than k skyband and potential-skybad elements will appear to be skyband elements. In this paper, since we focus on the problem of space usage estimation, we leave out the detailed implementation issues of the skyband algorithm.

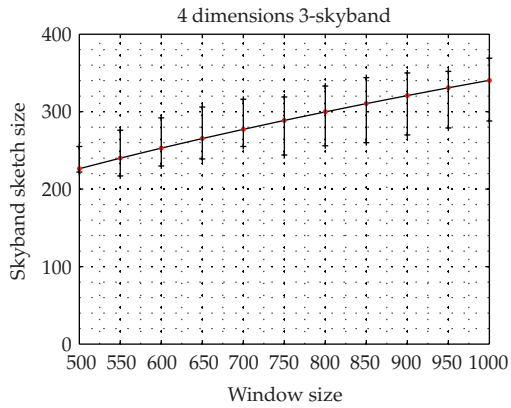
4. Space Usage Estimation

In this section, we present our robust approaches for estimating the space usage of sliding-window skybands under the assumption of statistical independence across dimensions based on the preliminary results in the previous section.

4.1. Distribution-Constrained Data

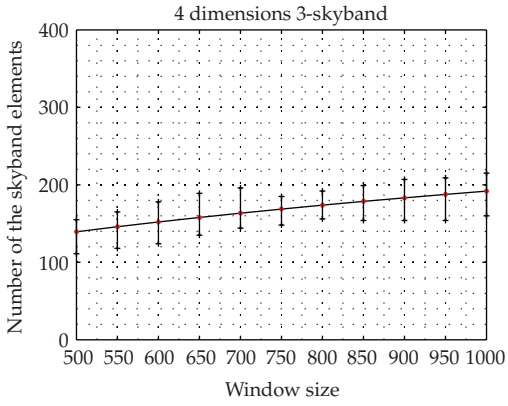
Here, we give our theoretical analysis for the space usage of sliding-window skybands over data which is distribution constrained, that is, there are no duplicate values over each dimension. By mapping the problem of evaluating the number of the elements in a finite set which satisfy no more than k of the m properties to the problem of evaluating the probability



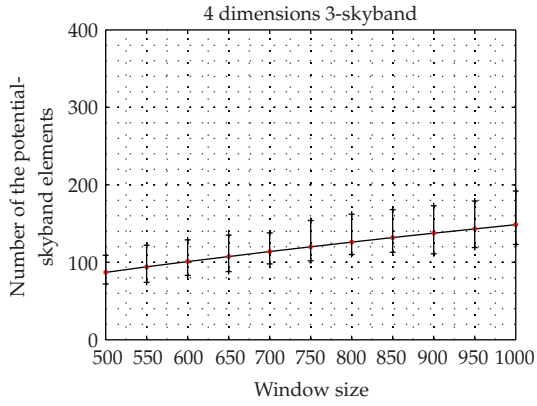


(a) Theoretical results

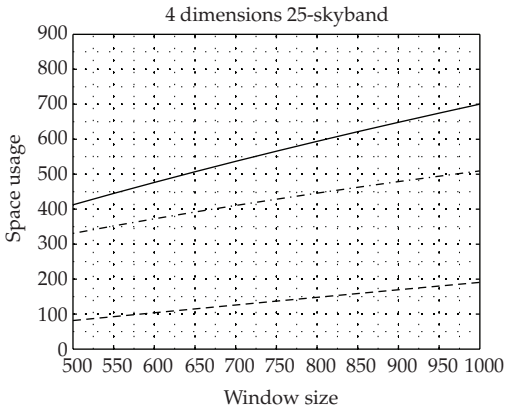
(b) Experimental results

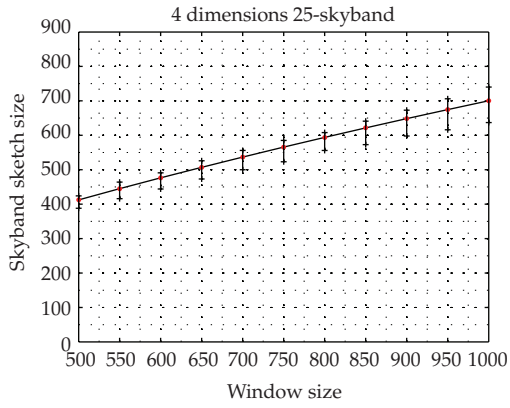


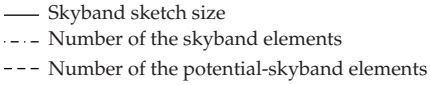
(c) Experimental results



(d) Experimental results



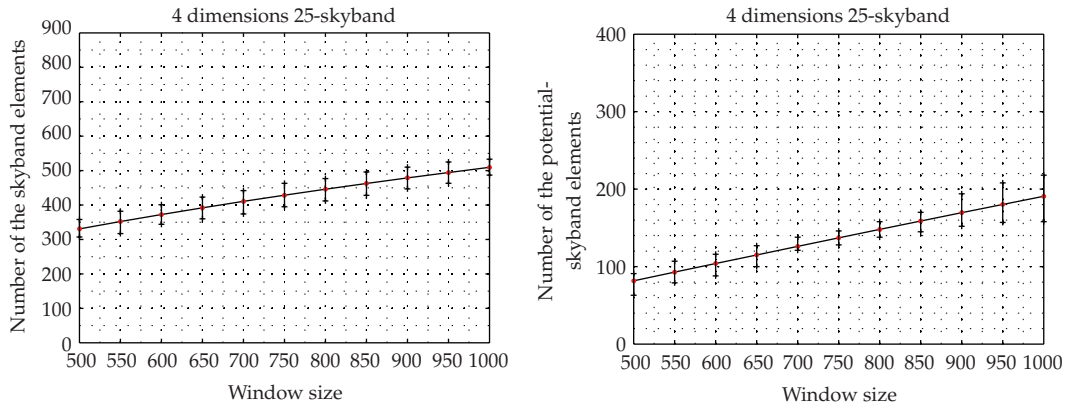




(e) Theoretical results

(f) Experimental results

Figure 3: Continued.



(g) Experimental results (h) Experimental results

Figure 3: Space performance of monitoring skylines over sliding windows in the stream environment in a 4-dimensional space where the data over each dimension is continuously distributed.

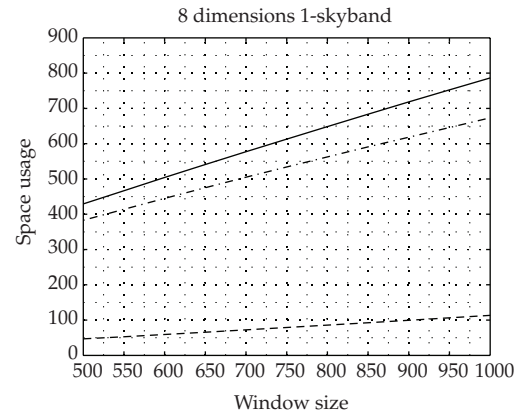
that no more than k of other m elements can dominate an element, Lemma 4.1 gives the probability that at most k of other m elements in a d-dimensional space can dominate an element. Based on Lemma 4.1, Theorem 4.2 gives the expected number of the k-skyband elements in a sliding window which contains n d-dimensional live elements.

Lemma 4.1. *Suppose that $\xi_0, \xi_1, \xi_2, \ldots, \xi_m$ are $m+1$ elements in a d-dimensional space, under assumptions of statistical independence across dimensions, no duplicate values over each dimension, and data domains being all totally ordered; let $D_k(m,d)$ be the fact that no more than k of other m elements can dominate ξ_0, then the probability of $D_k(m,d)$, that is, $\mathbb{P}\{D_k(m,d)\}$, can be characterized as*

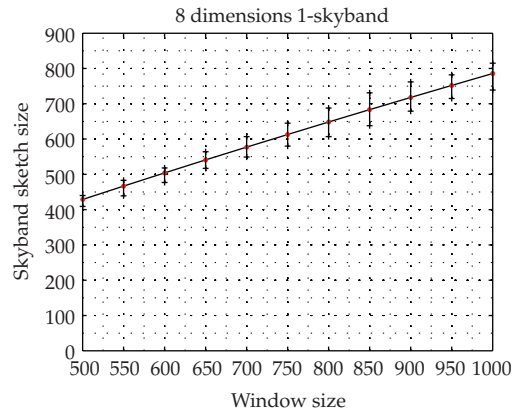
$$\mathbb{P}\{D_k(m,d)\} = 1 + \sum_{i=k+1}^{m} \frac{(-1)^{i-k}}{(i+1)^d} \binom{i-1}{k} \binom{m}{i}. \tag{4.1}$$

Proof. We map $\mathcal{S}$, $\mathcal{D}_i$, and $\mathcal{S}_i$ in Theorem 3.2 to the full probability space, $\xi_i \succ \xi_0$, and $\mathbb{P}\{\xi_i \succ \xi_0\}$, respectively; $\mathcal{T}(i)$ is mapped to $\mathcal{T}^\circ(i)$, which can be characterized as

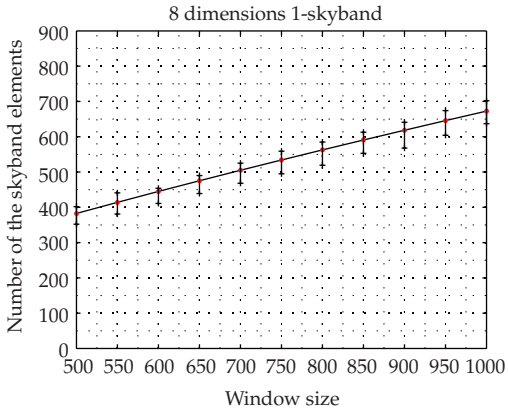
$$\begin{aligned}
\mathcal{T}^\circ(0) &= 1,\\
\mathcal{T}^\circ(1) &= \sum_{i=1}^{m} \mathbb{P}\{\xi_i \succ \xi_0\},\\
\mathcal{T}^\circ(2) &= \sum_{1\le i_1<i_2\le m} \mathbb{P}\{\xi_{i_1} \succ \xi_0 \wedge \xi_{i_2} \succ \xi_0\},\\
\mathcal{T}^\circ(3) &= \sum_{1\le i_1<i_2<i_3\le m} \mathbb{P}\{\xi_{i_1} \succ \xi_0 \wedge \xi_{i_2} \succ \xi_0 \wedge \xi_{i_3} \succ \xi_0\},\\
&\vdots\\
\mathcal{T}^\circ(m) &= \mathbb{P}\{\xi_1 \succ \xi_0 \wedge \xi_2 \succ \xi_0 \wedge \cdots \wedge \xi_m \succ \xi_0\}.
\end{aligned} \tag{4.2}$$



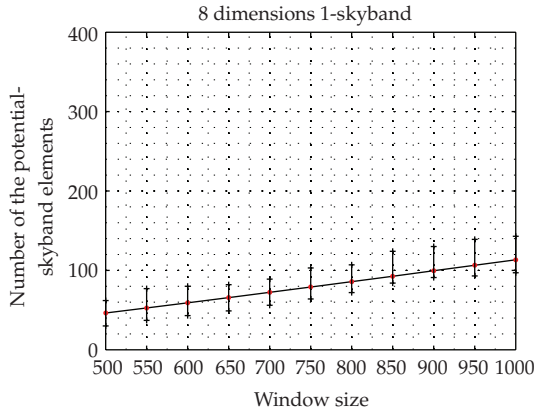
(a) Theoretical results



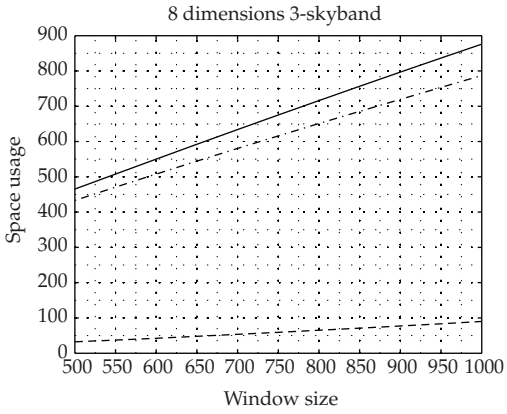
(b) Experimental results

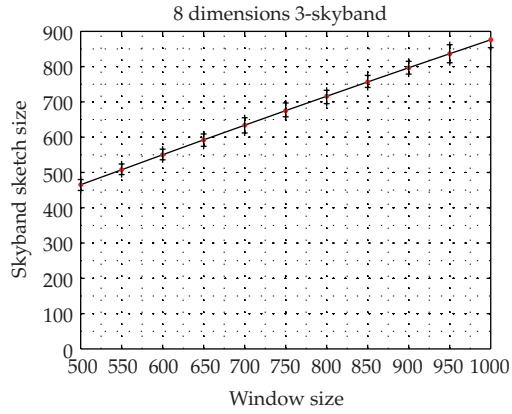


(c) Experimental results



(d) Experimental results



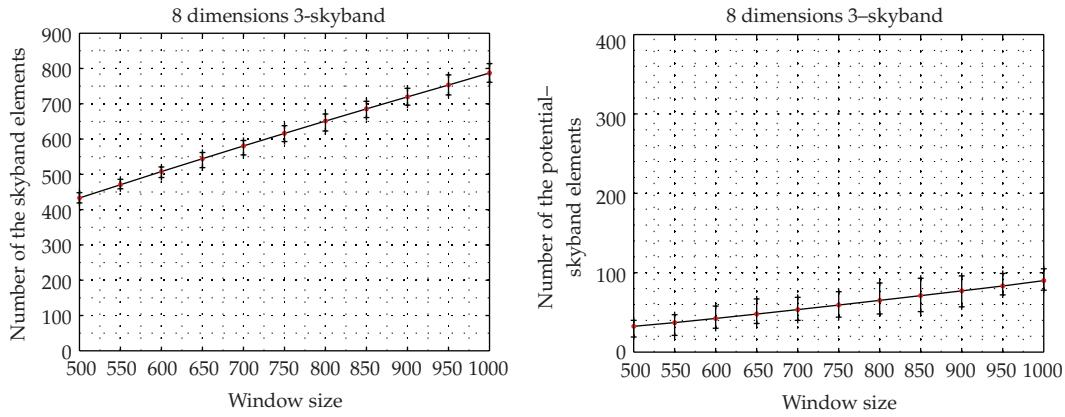


Skyband sketch size
Number of the skyband elements
Number of the potential-skyband elements

(e) Theoretical results

(f) Experimental results

Figure 4: Continued.



(g) Experimental results (h) Experimental results

Figure 4: Space performance of monitoring skylines over sliding windows in the stream environment in a 8-dimensional space where the data over each dimension is continuously distributed.

Under assumptions of statistical independence across dimensions, no duplicate values over each dimension, and domains being all totally-ordered, an element has a $1/(i+1)^d$ probability of being dominated by all other i elements; therefore, $\mathcal{T}^\circ(i)$ can be further characterized as

$$\mathcal{T}^\circ(i) = \frac{1}{(i+1)^d}\binom{m}{i}. \tag{4.3}$$

By Theorem 3.2, $\mathbb{P}\{D_k(m,d)\}$ can be characterized as

$$\begin{aligned}\mathbb{P}\{D_k(m,d)\} &= \mathcal{T}^\circ(0) + \sum_{i=k+1}^{m}(-1)^{i-k}\binom{i-1}{k}\mathcal{T}^\circ(i)\\ &= 1 + \sum_{i=k+1}^{m}\frac{(-1)^{i-k}}{(i+1)^d}\binom{i-1}{k}\binom{m}{i}.\end{aligned} \tag{4.4}$$

We have thus proved the lemma. □

Theorem 4.2. *Suppose that there are n d-dimensional live elements in a sliding window, under assumptions of statistical independence across dimensions, no duplicate values over each dimension, and dimension domains being all totally-ordered, the expected number of the k-skyband elements, that is, $\Psi_k(n,d)$, can be directly characterized as*

$$\Psi_k(n,d) = n + \sum_{i=k+1}^{n-1}\frac{(-1)^{i-k}}{(i+1)^{d-1}}\binom{i-1}{k}\binom{n}{i+1} \tag{4.5}$$

and can be recursively characterized as

$$\Psi_k(n,d) = \Psi_k(n-1,d) + \frac{\Psi_k(n,d-1)}{n} \tag{4.6}$$

with initial conditions $\Psi_k(n,1) = k+1$ *where* $n \geq k+1$ *and* $\Psi_k(k+1,d) = k+1$ *where* $d \geq 1$.

Proof. By Lemma 4.1, $\Psi_k(n,d)$ can be characterized as

$$\begin{aligned}
\Psi_k(n,d) &= n \cdot \mathbb{P}\{D_k(n-1,d)\} \\
&= n + n \cdot \sum_{i=k+1}^{n-1} \frac{(-1)^{i-k}}{(i+1)^d} \binom{i-1}{k} \binom{n-1}{i} \\
&= n + \sum_{i=k+1}^{n-1} \frac{(-1)^{i-k}}{(i+1)^{d-1}} \binom{i-1}{k} \binom{n}{i+1}.
\end{aligned} \tag{4.7}$$

$\Psi_k(n,d)$ can further be recursively characterized as

$$\begin{aligned}
\Psi_k(n,d) &= n + \sum_{i=k+1}^{n-1} \frac{(-1)^{i-k}}{(i+1)^{d-1}} \binom{i-1}{k} \binom{n}{i+1} \\
&= n + \sum_{i=k+1}^{n-2} \frac{(-1)^{i-k}}{(i+1)^{d-1}} \binom{i-1}{k} \binom{n-1}{i+1} \\
&\quad + \sum_{i=k+1}^{n-1} \frac{(-1)^{i-k}}{(i+1)^{d-1}} \binom{i-1}{k} \binom{n-1}{i} \\
&= (n-1) + \sum_{i=k+1}^{n-2} \frac{(-1)^{i-k}}{(i+1)^{d-1}} \binom{i-1}{k} \binom{n-1}{i+1} \\
&\quad + \frac{n}{n} + \frac{1}{n} \cdot \sum_{i=k+1}^{n-1} \frac{(-1)^{i-k}}{(i+1)^{d-2}} \binom{i-1}{k} \binom{n}{i+1} \\
&= \Psi_k(n-1,d) + \frac{\Psi_k(n,d-1)}{n}
\end{aligned} \tag{4.8}$$

with initial conditions

$$\begin{aligned}
&\Psi_k(n,1) = k+1 \ (n \geq k+1), \\
&\Psi_k(k+1,d) = k+1 \ (d \geq 1).
\end{aligned} \tag{4.9}$$

We have thus proved the theorem. □

Theorem 4.3 shows that there exists inherent correlation between the expected number of the skyband elements in case of monitoring a $(d+1)$-dimensional k-skyband over a sliding

window which contains n elements $\Psi_k(n, d+1)$ and the expected number of the elements stored by the skyband sketch in case of monitoring a d-dimensional k-skyband over a sliding window which contains n elements $\Phi_k(n, d)$, that is, $\Psi_k(n, d+1) = \Phi_k(n, d)$. In addition, the expected number of the potential-skyband elements in case of monitoring a d-dimensional k-skyband over a sliding window which contains n elements $\Omega_k(n, d)$ equals $\Phi_k(n, d) - \Psi_k(n, d)$. Therefore, by a minor revision, Theorem 4.2 can also be used to characterize the expected number of the potential-skyband elements.

Theorem 4.3. *Under assumptions of statistical independence across dimensions, no duplicate values over each dimension, and domains being all totally-ordered, the expected number of the skyband elements in case of monitoring a $(d+1)$-dimensional k-skyband over a sliding window which contains n live elements, that is, $\Psi_k(n, d+1)$, equals the expected number of the elements stored by the skyband sketch in case of monitoring a d-dimensional k-skyband over a sliding window which contains n live elements, that is, $\Phi_k(n, d)$.*

Proof. By Lemma 4.1, $\Phi_k(n, d)$ can be characterized as

$$\begin{aligned}
\Phi_k(n,d) &= k + 1 + \sum_{j=k+2}^{n} \mathbb{P}\{D_k(j-1,d)\} \\
&= n + \sum_{j=k+2}^{n} \sum_{i=k+1}^{j-1} \frac{(-1)^{i-k}}{(i+1)^d} \binom{i-1}{k} \binom{j-1}{i} \\
&= n + \sum_{i=k+1}^{n-1} \sum_{j=i+1}^{n} \frac{(-1)^{i-k}}{(i+1)^d} \binom{i-1}{k} \binom{j-1}{i} \\
&= n + \sum_{i=k+1}^{n-1} \left(\frac{(-1)^{i-k}}{(i+1)^d} \binom{i-1}{k} \sum_{j=i}^{n-1} \binom{j}{i} \right) \\
&= n + \sum_{i=k+1}^{n-1} \frac{(-1)^{i-k}}{(i+1)^d} \binom{i-1}{k} \binom{n}{i+1} \\
&= \Psi_k(n, d+1).
\end{aligned} \tag{4.10}$$

To see why the theorem holds, suppose $\xi_1, \xi_2, \ldots, \xi_n$ are the n live elements in the sliding window, which are ascendingly ordered by the element sequence number, and $\xi_{i_1}, \xi_{i_2}, \ldots, \xi_{i_m}$, where $1 \le i_1 < i_2 < \cdots < i_m \le n$, are the m elements stored by the skyband sketch for monitoring a d-dimensional k-skyband over the sliding window. We map each of the live element $\xi_i = \langle x_{i1}, x_{i2}, \ldots, x_{id} \rangle$ into a $(d+1)$-dimensional elements $\xi_i^\circ = \langle x_{i1}, x_{i2}, \ldots, x_{id}, 1/\eta_i \rangle$, where η_i is the sequence number of the element, then $\xi_{i_1}^\circ, \xi_{i_2}^\circ, \ldots, \xi_{i_m}^\circ$ are just the k-skyband elements in the $(d+1)$-dimensional space. □

4.2. A Dynamic Programming Algorithm

In this subsection, based on the theoretical analysis proposed in the above subsection, we propose an efficient dynamic programming algorithm to estimate the space usage. Since there

```
Input: n: the number of the elements
       d: the number of the dimensions
       k: the k-skyband
Output: the expected skyband cardinality
begin
  if n ≤ k + 1 then return n;
  if d = 1 then return k + 1;
  for i = 1 to d do α[i] ← k + 1;
  ξ ← 0;
  for i = k + 2 to n do
      ξ ← (ξ + 1) mod 2;
      if ξ = 1 then
         β[1] ← k + 1;
         for j = 2 to d do β[j] ← β[j − 1]/i + α[j];
      else
         α[1] ← k + 1;
         for j = 2 to d do α[j] ← α[j − 1]/i + β[j];
      end
  end
  if ξ = 0 then return α[d] else return β[d];
end
```

Algorithm 1: Expected skyband cardinality: $\Psi_k(n, d)$.

exist inherent correlations among the expected number of the skyband elements, the expected number of the potential-skyband elements, and the expected number of the elements stored by the skyband sketch, we only consider how to estimate the number of the skyband elements.

Estimating the number of the skyband elements using (4.5) is infeasible in most cases because combination numbers are used to characterize the expected number of the skyband elements; for example, the number of the different ways of selecting 50 elements from 100 different elements can not be stored by a 64-bit integer. Based on (4.6), we can design a recursive algorithm to estimate the number of the skyband elements, which will not encounter integer overflow. The recursive algorithm can be characterized by a binary tree with the depth of $\max(n - k, d)$, where n, k, and d are the same as those in Theorem 4.2. Therefore, estimating the number of the skyband elements using the recursive algorithm has the computational complexity of $\Theta(2^{\max(n-k,d)})$, which is unacceptable in most cases. Actually, there exists a large amount of duplicate computations in the binary tree; therefore, if duplicate computations can be eliminated, the computational complexity can be reduced. Algorithm 1 is a nonrecursive algorithm for estimating the number of the skyband elements, which is based on (4.6), and all the duplicate computations are eliminated. The algorithm is a dynamic programming algorithm [33], because although the algorithm is based on a recurrence, it is non-recursive, and each step of the algorithm gives an exact answer for the corresponding subproblem.

Algorithm 1 functions as follows. First, two vectors α and β with size d are created, and the values of $\alpha[1 \cdots d]$ are initialized to $\Psi_k(k+1, 1 \cdots d)$, respectively. According to the initial conditions, we have $\Psi_k(k+1, 1 \cdots d) = k+1$, hence all the values of $\alpha[1 \cdots d]$ are initialized to $k+1$. Then, we evaluate $\Psi_k(k+2, 1 \cdots d)$ and store the values to $\beta[1 \cdots d]$, respectively. According to the initial conditions, we have $\Psi_k(k+2, 1) = k+1$, hence $\beta[1]$ is set to $k+1$.

According to the recurrence, we have $\Psi_k(k+2,2) = \Psi_k(k+2,1)/(k+2) + \Psi_k(k+1,2)$, that is, $\beta[2] = \beta[1]/(k+2) + \alpha[2]$, hence we can evaluate $\Psi_k(k+2,2)$ and store the value to $\beta[2]$. By the same principle, we may evaluate $\Psi_k(k+2,3\cdots d)$ sequentially and store the values to $\beta[3\ldots d]$. We may continue to evaluate $\Psi_k(k+3,1\cdots d)$ using the values in $\beta[1\cdots d]$ and store the values of $\Psi_k(k+3,1\cdots d)$ to $\alpha[1\ldots d]$, respectively, until we evaluate $\Psi_k(n,1\cdots d)$ and store the values to $\alpha[1\cdots d]$ or $\beta[1\cdots d]$. At last, the value of $\alpha[d]$ or $\beta[d]$ is returned as the value of $\Psi_k(n,d)$. It is apparent that the algorithm is space and time efficient, because the space complexity and the time complexity are $\Theta(d)$ and $\Theta((n-k)d)$, respectively.

5. Experiments

In this section, we verify our theoretical results on space usage estimation of the k-skyband operator monitoring skybands over sliding windows in the stream environment by extensive experiments. The algorithms have been implemented by the C++ programming language and run on a 2.0 GHz Intel CPU with 2 GB of memory, and the data over each dimension is generated by the (GNU Scientific Library GSL: http://www.gnu.org/software/gsl). We test the space performance in a lower dimensional (4-dimensional) and a higher dimensional (8-dimensional) space, respectively. According to the probability theory, if the data over a dimension is continuously distributed, the probability that there are duplicate values over the dimension is zero. Therefore, for each space, we generate a dataset; the data over the first dimension is normally distributed with $\sigma = 500$, and the data over other dimensions is normally distributed with $\sigma = 100$. At the same time, the sliding-window size increases from 500 to 1000 stepped by 50; for each step, we compute the maximal, average, and minimal skyband sketch size, number of the skyband elements, and number of the potential-skyband elements during the moving of the sliding window over one million elements. Since there is no previous work that evaluates the space usage over continuous data, thus we compare our corresponding theoretical results with the experimental results.

Figures 3 and 4 show the comparisons between experimental results and the theoretical results for 4-dimension space and 8-dimension space. We can see that the experimental results are almost the same as we expected in the theories. What is more is that the maximal values are not twice as much as the minimal value and they are all close to the theoretical results. For the given parament r (r-skyband) and d, both of the actual space usage and the estimated space usage increase with the window size, as more objects need to be evaluated. At the same time, the skyband cardinality also increases when the value of parament r increases. The comparison between 4-dimension space and 8-dimension space, as Figures 3(a) and 4(e) show, illustrates that the skyband sketch size in high-dimension space is much more than that in low-dimension space, when the window size and the parament r are given. This is because less elements are likely to be dominated by other objects in high-dimension space compared with in low-dimension space. As there are sufficient skylines for users to make a decision in the higher-dimensional space, skybands query shows its efficiency in low-dimensional space.

6. Conclusions and Discussions

Skyband query is of great importance for multi-criteria decision-making applications. To support skyband query in the stream engine, the problem of effective space usage estimation must be solved, which is important for extending the query optimizers cost model. In this

paper, under the assumption of statistically independent [34, 35] across dimensions, no duplicate values over each dimension, and dimension domains being all totally ordered, we propose effective methods to address this issue; since the skyline query is just a special case of skyband queries, it is obvious that our approaches apply to sliding-window skyline queries either. We also put forward a dynamic programming algorithm to estimate the space usage, which is space and time efficient. In addition, if only the distribution function is given, we can also use the similar approach to evaluate the skyband cardinality over a space, where there are duplicate values over some dimensions. Finally, we carried out extensive experiments which verified that our proposed approaches can estimate the space usage accurately, hence, can be used to extend the optimizer's cost model for incorporating the skyband operator.

Acknowledgments

This work is partially supported by China "863" Hi-tech Program (Grant no. 2007AA01Z153), Zhejiang Provincial NSF (Grant no. Y1090096), and the National Natural Science Foundation of China (NSFC) under Grant no. 60573125 and 60873264.

References

[1] S. Börzsönyi, D. Kossmann, and K. Stocker, "The skyline operator," in *Proceedings of the International Conference on Data Engineering (ICDE '01)*, pp. 421–430, 2001.

[2] D. Papadias, Y. Tao, G. Fu, and B. Seeger, "Progressive skyline computation in database systems," *ACM Transactions on Database Systems*, vol. 30, no. 1, pp. 41–82, 2005.

[3] C. Buchta, "On the average number of maxima in a set of vectors," *Information Processing Letters*, vol. 33, no. 2, pp. 63–65, 1989.

[4] X. Lin, Y. Yuan, W. Wang, and H. Lu, "Stabbing the sky: efficient skyline computation over sliding windows," in *Proceedings of the International Conference on Data Engineering (ICDE '05)*, pp. 502–513, 2005.

[5] Y. Tao and D. Papadias, "Maintaining sliding window skylines on data streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 3, pp. 377–391, 2006.

[6] J. Chomicki, P. Godfrey, J. Gryz, and D. Liang, "Skyline with presorting," in *Proceedings of the International Conference on Data Engineering (ICDE '03)*, pp. 717–719, 2003.

[7] P. Godfrey, R. Shipley, and J. Gryz, "Maximal vector computation in large data sets," in *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB '05)*, vol. 1, pp. 229–240, 2005.

[8] D. Kossmann, F. Ramsak, and S. Rost, "Shooting stars in the sky: an online algorithm for skyline queries," in *Proceedings of the International Conference on Very Large Data Bases (VLDB '02)*, pp. 275–286, 2002.

[9] D. Papadias, Y. Tao, G. Fu, and B. Seeger, "An optimal and progressive algorithm for skyline queries," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '03)*, pp. 467–478, 2003.

[10] K.-L. Tan, P.-K. Eng, and B. C. Ooi, "Efficient progressive skyline computation," in *Proceedings of the International Conference on Very Large Data Bases (VLDB '01)*, pp. 301–310, 2001.

[11] C.-Y. Chan, P.-K. Eng, and K.-L. Tan, "Efficient processing of skyline queries with partially-ordered domains," in *Proceedings of the International Conference on Data Engineering (ICDE '05)*, pp. 190–191, 2005.

[12] M. Morse, J. M. Patel, and H. V. Jagadish, "Efficient skyline computation over low-cardinality domains," in *Proceedings of the International Conference on Very Large Data Bases (VLDB '07)*, pp. 267–278, 2007.

[13] Y. Tao, K. Xiao, and J. Pei, "SUBSKY: efficient computation of skylines in subspaces," in *Proceedings of the International Conference on Data Engineering (ICDE '06)*, p. 65, 2006.

[14] A. Vlachou, C. Doulkeridis, Y. Kotidis, and M. Vazirgiannis, "SKYPEER: efficient subspace skyline computation over distributed data," in *Proceedings of the International Conference on Data Engineering (ICDE '07)*, pp. 416–425, 2007.

[15] C. Li, B. C. Ooi, A. K. H. Tung, and S. Wang, "DADA: a data cube for dominant relationship analysis," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '06)*, pp. 659–670, 2006.

[16] J. Pei, A.W.-C. Fu, X. Lin, and H. Wang, "Computing compressed multidimensional skyline cubes efficiently," in *Proceedings of the International Conference on Data Engineering (VLDB '07)*, pp. 96–105, 2007.

[17] J. Pei, W. Jin, M. Ester, and Y. Tao, "Catching the best views of skyline: a semantic approach based on decisive subspaces," in *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB '05)*, vol. 1, pp. 253–264, 2005.

[18] T. Xia and D. Zhang, "Refreshing the sky: the compressed skycube with efficient support for frequent updates," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '06)*, pp. 491–502, 2006.

[19] Y. Yuan, X. Lin, Q. Liu, W. Wang, J. X. Yu, and Q. Zhang, "Efficient computation of the skyline cube," in *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB '05)*, vol. 1, pp. 241–252, 2005.

[20] W.-T. Balke, U. Güntzer, and J. X. Zheng, "Efficient distributed skylining for web information systems," in *Proceedings of the 9th International Conference on Extending Database Technology (EDBT '04)*, pp. 256–273, 2004.

[21] E. Lo, K. Y. Yip, K.-I. Lin, and D. W. Cheung, "Progressive skylining over Web-accessible databases," *Data and Knowledge Engineering*, vol. 57, no. 2, pp. 122–147, 2006.

[22] S. Wang, B. C. Ooi, A. K.H. Tung, and L. Xu, "Efficient skyline query processing on peer-to-peer networks," in *Proceedings of the International Conference on Data Engineering (ICDE '07)*, pp. 1126–1135, 2007.

[23] P. Wu, C. Zhang, Y. Feng, B. Y. Zhao, D. Agrawal, and A. El Abbadi, "Parallelizing skyline queries for scalable distribution," in *Proceedings of the 10th International Conference on Extending Database Technology (EDBT '06)*, pp. 112–130, 2006.

[24] C.-Y. Chan, H. V. Jagadish, K.-L. Tan, A. K. H. Tung, and Z. Zhang, "Finding k-dominant skylines in high dimensional space," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '06)*, pp. 503–514, 2006.

[25] C.-Y. Chan, H. V. Jagadish, K.-L. Tan, A. K. H. Tung, and Z. Zhang, "On high dimensional skylines," in *Proceedings of the 10th International Conference on Extending Database Technology (EDBT '06)*, pp. 478–495, 2006.

[26] E. Dellis and B. Seeger, "Efficient computation of reverse skyline queries," in *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB '07)*, pp. 291–302, 2007.

[27] X. Lin, Y. Yuan, Q. Zhang, and Y. Zhang, "Selecting stars: the k most representative skyline operator," in *Proceedings of the International Conference on Data Engineering (ICDE '07)*, pp. 86–95, 2007.

[28] J. Pei, B. Jiang, X. Lin, and Y. Yuan, "Probabilistic skylines on uncertain data," in *Proceedings of the International Conference on Very Large Data Bases (VLDB '07)*, pp. 15–26, 2007.

[29] J. L. Bentley, H. T. Kung, M. Schkolnick, and C. D. Thompson, "On the average number of maxima in a set of vectors and applications," *Journal of the Association for Computing Machinery*, vol. 25, no. 4, pp. 536–543, 1978.

[30] P. Godfrey, "Skyline cardinality for relational processing: how many vectors are maximal?" in *Proceedings of the 3rd International Symposium Foundations of Information and Knowledge Systems (FoIKS '04)*, pp. 78–97, 2004.

[31] S. Chaudhuri, N. Dalvi, and R. Kaushik, "Robust cardinality and cost estimation for skyline operator," in *Proceedings of the International Conference on Data Engineering (ICDE '06)*, p. 64, 2006.

[32] K. H. Rosen, *Discrete Mathematics and Its Applications*, WCB/McGraw-Hill, Boston, Mass, USA, 4th edition, 1999.

[33] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, Cambridge, Mass, USA, 2nd edition, 2001.

[34] M. Li, "Fractal time series—a tutorial review," *Mathematical Problems in Engineering*, vol. 2010, Article ID 157264, 26 pages, 2010.

[35] M. Li and W. Zhao, "Representation of a stochastic traffic bound," *IEEE Transactions on Parallel and Distributed Systems*, 2009.