Effective Skyline Cardinality Estimation on Data Streams*

Yang Lu, Jiakui Zhao, Lijun Chen, Bin Cui, and Dongqing Yang

Key Laboratory of High Confidence Software Technologies (Peking University), Ministry of Education, China
School of Electronics Engineering and Computer Science, Peking University, China
{yanglu,jkzhao,ljchen,bin.cui,dqyang}@pku.edu.cn

Abstract. In order to incorporate the skyline operator into the data stream engine, we need to address the problem of skyline cardinality estimation, which is very important for extending the query optimizer's cost model to accommodate skyline queries. In this paper, we propose robust approaches for estimating the skyline cardinality over sliding windows in the stream environment. We first design an approach to estimate the skyline cardinality over uniformly distributed data, and then extend the approach to support arbitrarily distributed data. Our approaches allow arbitrary data distribution, hence can be applied to extend the optimizer's cost model. To estimate the skyline cardinality in online manner, the live elements in the sliding window are sketched using Spectral Bloom Filters which can efficiently and effectively capture the information which is essential for estimating the skyline cardinality over sliding windows. Extensive experimental study demonstrates that our approaches significantly outperform previous approaches.

1 Introduction

Skyline queries are very important for many applications, such as data mining and multi-criteria decision making, and have attracted much attention [4,9,12]. Given two multi-dimensional elements ξ_1 and ξ_2, if ξ_1 is better than or equal to ξ_2 over all dimensions and strictly better than ξ_2 over at least one dimension, we say that ξ_1 dominates ξ_2, and is marked as $\xi_1 \succ \xi_2$. If an element is not dominated by any other element, it is a skyline element, and the skyline query returns all skyline elements. Continuously monitoring skylines over sliding windows [11,14] in the stream environment also received much attention; the skyline changes over time as the window slides and the skyline changes are reported to the user continuously in real-time manner. In order to incorporate the skyline operator into the data stream engine, we need to solve the problem of skyline cardinality estimation, which is very important for extending the optimizer's cost model.

There are some previous works [2,5] which considered the problem of skyline cardinality estimation over static datasets. However, the approaches are based

* This work is supported by project 2007AA01Z153 under the National High-tech Research and Development of China and the National Natural Science Foundation of China under Grant No.60603045.

S.S. Bhowmick, J. Küng, and R. Wagner (Eds.): DEXA 2008, LNCS 5181, pp. 241–254, 2008.

on very strong assumptions on the data distribution, e.g., no duplicate values over each dimension. The approach in [6] allows duplicate values, but only two possible values, e.g. 0 and 1, are allowed over the dimension which contains duplicate values, hence the restriction is still very strong. Since duplicate values are very common, above approaches do not scale well to real-life applications. In this paper, we propose robust approaches for estimating the skyline cardinality, and our approaches can support the skyline computation over arbitrarily distributed data. In addition, since the elements in the sliding window change over time as the window slides, we use Spectral Bloom Filters [7] to continuously capture the information which is essential for estimating the skyline cardinality. Our contributions in this paper can be summarized as follows:

1. We propose an approach which only uses the value cardinality of each dimension to estimate the skyline cardinality, under the assumption that the data over each dimension is uniformly distributed.
2. We design a robust approach which considers the data distribution over each dimension. This enhanced approach can estimate the skyline cardinality effectively and efficiently over arbitrarily distributed data.
3. We propose to use Spectral Bloom Filters to capture the information, such as value cardinality and value frequency over each dimension, which is essential for estimating the skyline cardinality over sliding windows.
4. We conduct extensive experimental study to demonstrate that our approaches yield better performance than existing approaches.

The rest of this paper is organized as follows: Section 2 surveys related works; Section 3 proposes our skyline cardinality estimation approaches; experimental results are shown in Section 4, followed by our conclusion in Section 5.

2 Related Works

The skyline problem was originally studied as the maximal vector problem; Kung et al. [10] proposed the first algorithm for finding the maximal vectors from a set of memory resident vectors. Börzsönyi et al. [4] introduced the skyline operator into relational database systems. Recently, continuously monitoring skylines over sliding windows [11,14] also received much attention.

There are some previous works which considered the problem of skyline cardinality estimation over static datasets. Under assumptions of statistical independence across dimensions, no duplicate values over each dimension, and dimension domains are all totally ordered, Bentley et al. [2] and Godfrey [8] proposed methods for estimating the skyline cardinality using carefully designed recurrence. Under the same assumptions, Buchta [5] and Chaudhuri et al. [6] proposed to estimate the skyline cardinality using integrals, and Buchta [5] further derived that the skyline cardinality equals $\Theta\left((\ln n)^{k-1}/(k-1)!\right)$, where n is the number of elements in the space and k is the number of dimensions. Above approaches cannot be applied to most real-life applications as the "no duplicate values" assumption is impractical. Chaudhuri et al. [6] relaxed the "no duplicate values"

assumption, but the dimensions which contain duplicate values can only have two possible values, which is still a suffering constraint. In addition, the approach uses integrals to estimate the skyline cardinality, and the integrals have no a close form, hence can only be approximated by some scientific computing methods. In this paper, under assumptions of statistical independence across dimensions which is commonly used by query optimizers, we consider the problem of sliding-window skyline cardinality estimation over arbitrarily distributed data in the stream environment.

3 Skyline Cardinality Estimation

In this section, we introduce how to estimate the skyline cardinality over sliding windows in the stream environment.

3.1 Estimation under Strong Assumptions

The approaches for skyline cardinality estimation proposed in [2,5,6] can be extended to the stream context without modifications and are summarized by Theorem 1, Theorem 2 and Theorem 3 respectively. Theorem 1 and Theorem 2 are theoretically equivalent. The three approaches suffer from the strong assumptions as described in Section 2, and hence do not apply to real-life applications.

Theorem 1. *Suppose that there are n k-dimensional live elements in a sliding window; under assumptions of statistical independence across dimensions, no duplicate values over each dimension, and dimension domains are all totally ordered, the expected number of the skyline elements $\Psi(n,k)$ can be recursively characterized as*

$$\Psi(n,k) = \Psi(n-1,k) + \frac{1}{n}\Psi(n,k-1)$$

with initial conditions

$$\begin{aligned} \Psi(1,k) &= 1 \qquad k \geq 1 \\ \Psi(n,1) &= 1 \qquad n \geq 1. \end{aligned}$$

Theorem 2. *Suppose that there are n k-dimensional live elements in a sliding window; under the same assumptions as those in Theorem 1, the expected number of the skyline elements $\Psi(n,k)$ can be characterized as*

$$\Psi(n,k) = n \int_0^1 \cdots \int_0^1 (1 - x_1 \cdots x_k)^{n-1} \mathrm{d}x_1 \cdots \mathrm{d}x_k.$$

Theorem 3. *Suppose that there are n k-dimensional live elements in a sliding window; there are only two possible values over each of the first k° dimensions, and there are no duplicate values over the other dimensions. For simplicity and without loss of generality, suppose that the two possible values over the first k° dimensions are 1 and 0, and 1 dominates 0; p_i denotes the probability that the*

value of the ith dimension equals 1*, where* $1 \leq i \leq k^\circ$*. Under assumptions of statistical independence across dimensions and dimension domains are all totally ordered, the expected number of the skyline elements* $\Psi(n,k)$ *equals*

$$n \sum_{v \in \{0,1\}^{k^\circ}} \int_0^1 \cdots \int_0^1 \Phi_v(x_1, \cdots, x_{k-k^\circ}) \mathrm{d}x_1 \cdots \mathrm{d}x_{k-k^\circ}$$

where $\Phi_v(x_1, \cdots, x_{k-k^\circ})$ *can be characterized as*

$$\Phi_v(x_1, \cdots, x_{k-k^\circ}) = \mathbb{P}_1(v) \left(1 - \mathbb{P}_2(v) x_1 \cdots x_{k-k^\circ}\right)^{n-1}$$

$$\mathbb{P}_1(v) = \prod_{i=1}^{k^\circ} p_i^{v_i} (1-p_i)^{1-v_i}$$

$$\mathbb{P}_2(v) = \prod_{i=1}^{k^\circ} p_i^{1-v_i}.$$

3.2 Estimation over Uniformly Distributed Data

As we introduced previously, existing approaches assume that there are no duplicate values over each dimension or the dimension only has two possible values. To achieve better applicability, we relax the restriction on the data distribution. Under assumptions of statistical independence across dimensions and the data over each dimension is uniformly distributed, we use the value cardinality, i.e. the number of the distinct elements, over each dimension to characterize the expected number of the skyline elements. The theoretical analysis is based on the *Inclusion-Exclusion Principle* [13]. Lemma 1 gives the probability that an element is dominated by all other n elements. Lemma 2 gives the probability that an element is dominated by at least one of other n elements. Lemma 3 gives the probability that none of other n elements can dominate an element. Theorem 4 gives the expected number of the skyline elements in a sliding window which contains n k-dimensional live elements.

Lemma 1. *Suppose that* $\xi_0 \xi_1 \cdots \xi_n$ *are* $n+1$ *k-dimensional elements, where* $\xi_i =< x_{i1}, x_{i2}, \cdots, x_{ik} >$*. The data over each dimension is uniformly distributed, and the value cardinality of the* j*th dimension, i.e. the number of the distinct values over the* j*th dimension, is* c_j*;* $v_{j1}, v_{j2}, \cdots, v_{jc_j}$ *denote the distinct values over the* j*th dimension, where* $v_{j1} < v_{j2} < \cdots < v_{jc_j}$*. Under assumptions of statistical independence across dimensions, the probability that* $\forall_{i(1 \leq i \leq n)}(\xi_i \succ \xi_0)$*, i.e.* $\mathbb{P}_\circ(n)$*, can be characterized as,*

$$\mathbb{P}_\circ(n) = \prod_{j=1}^{k} \sum_{t=1}^{c_j} \frac{t^n}{c_j^{n+1}} \qquad n \geq 1,\ k \geq 1.$$

Proof. $\mathbb{P}_\circ(n)$ can be characterized as,

$$\mathbb{P}_\circ(n) = \mathbb{P}\{(\forall_{i(1 \leq i \leq n)}(\xi_i \succ \xi_0)\}$$

$$= \prod_{j=1}^{k} \sum_{t=1}^{c_j} \mathbb{P}\{x_{0j} = v_{jt}\} \mathbb{P}\{\forall_{i(1\le i\le n)}(x_{0j} \ge x_{ij}) \mid x_{0j} = v_{jt}\}$$
$$= \prod_{j=1}^{k} \sum_{t=1}^{c_j} \left(\frac{1}{c_j} \cdot \left(\sum_{\theta=1}^{t} \frac{1}{c_j} \right)^n \right) = \prod_{j=1}^{k} \sum_{t=1}^{c_j} \frac{t^n}{c_j^{n+1}}.$$

Lemma 2. *Under the same conditions as those in Lemma 1, the probability that* $\exists_{i(1\le i\le n)}(\xi_i \succ \xi_0)$*, i.e.* $\mathbb{P}_\bullet(n)$*, can be characterized as,*

$$\mathbb{P}_\bullet(n) = \sum_{i=1}^{n} \left((-1)^{i-1} \binom{n}{i} \prod_{j=1}^{k} \sum_{t=1}^{c_j} \frac{t^i}{c_j^{i+1}} \right) \quad n \ge 1,\ k \ge 1.$$

Proof. Using the *Inclusion-Exclusion Principle* [13] and Lemma 1, we have

$$\begin{aligned}
\mathbb{P}_\bullet(n) &= \mathbb{P}\{(\xi_1 \succ \xi_0) \vee \cdots \vee (\xi_n \succ \xi_0)\} \\
&= \sum_{i=1}^{n} \mathbb{P}\{\xi_i \succ \xi_0\} - \sum_{1\le i_1<i_2\le n} \mathbb{P}\{(\xi_{i_1} \succ \xi_0) \wedge (\xi_{i_2} \succ \xi_0)\} \\
&\quad + \sum_{1\le i_1<i_2<i_3\le n} \mathbb{P}\{(\xi_{i_1} \succ \xi_0) \wedge (\xi_{i_2} \succ \xi_0) \wedge (\xi_{i_3} \succ \xi_0)\} \\
&\quad - \cdots + (-1)^{n-1} \mathbb{P}\{(\xi_1 \succ \xi_0) \wedge \cdots \wedge (\xi_n \succ \xi_0)\} \\
&= \binom{n}{1} \mathbb{P}_\circ(1) - \binom{n}{2} \mathbb{P}_\circ(2) + \binom{n}{3} \mathbb{P}_\circ(3) - \cdots + (-1)^{n-1} \binom{n}{n} \mathbb{P}_\circ(n) \\
&= \sum_{i=1}^{n} (-1)^{i-1} \binom{n}{i} \mathbb{P}_\circ(i) = \sum_{i=1}^{n} \left((-1)^{i-1} \binom{n}{i} \prod_{j=1}^{k} \sum_{t=1}^{c_j} \frac{t^i}{c_j^{i+1}} \right).
\end{aligned}$$

Lemma 3. *Under the same conditions as those in Lemma 1, the probability that* $\nexists_{i(1\le i\le n)}(\xi_i \succ \xi_0)$*, i.e.* $\mathbb{P}_\star(n)$*, can be characterized as,*

$$\mathbb{P}_\star(n) = \sum_{t_1=1}^{c_1} \cdots \sum_{t_k=1}^{c_k} \left(\prod_{i=1}^{k} \frac{1}{c_i} \right) \left(1 - \prod_{j=1}^{k} \frac{t_j}{c_j} \right)^n \quad n \ge 1,\ k \ge 1.$$

Proof. By Lemma 2, $\mathbb{P}_\star(n)$ can be characterized as,

$$\begin{aligned}
&\mathbb{P}_\star(n) = \mathbb{P}\{\nexists_{i(1\le i\le n)}(\xi_i \succ \xi_0)\} = 1 - \mathbb{P}_\bullet(n) \\
&= \sum_{i=0}^{n} \left((-1)^i \binom{n}{i} \prod_{j=1}^{k} \sum_{t=1}^{c_j} \frac{t^i}{c_j^{i+1}} \right) = \sum_{t_1=1}^{c_1} \cdots \sum_{t_k=1}^{c_k} \left(\prod_{l=1}^{k} \frac{1}{c_l} \right) \sum_{i=0}^{n} (-1)^i \binom{n}{i} \prod_{j=1}^{k} \frac{t_j^i}{c_j^i} \\
&= \sum_{t_1=1}^{c_1} \cdots \sum_{t_k=1}^{c_k} \left(\prod_{l=1}^{k} \frac{1}{c_l} \right) \sum_{i=0}^{n} \binom{n}{i} \left(-\prod_{j=1}^{k} \frac{t_j}{c_j} \right)^i = \sum_{t_1=1}^{c_1} \cdots \sum_{t_k=1}^{c_k} \left(\prod_{i=1}^{k} \frac{1}{c_i} \right) \left(1 - \prod_{j=1}^{k} \frac{t_j}{c_j} \right)^n.
\end{aligned}$$

Theorem 4. *Suppose that there are n k-dimensional live elements in the sliding window; the data over each dimension is uniformly distributed, and the value cardinality of the jth dimension is c_j; under the assumption of statistical independence across dimensions, the expected number of the skyline elements $\Psi(n,k)$ can be characterized as,*

$$\Psi(n,k) = n \cdot \sum_{t_1=1}^{c_1} \cdots \sum_{t_k=1}^{c_k} \left(\prod_{i=1}^{k} \frac{1}{c_i} \right) \left(1 - \prod_{j=1}^{k} \frac{t_j}{c_j} \right)^{n-1}$$

where $n \geq 1$ and $k \geq 1$.

Proof. By Lemma 3 and $\Psi(n,k) = n\mathbb{P}_\star(n-1)$, the theorem can be easily proved.

3.3 Estimation over Arbitrarily Distributed Data

Theorem 4 assumes that the data over each dimension is uniformly distributed; however, skewed data is very common in real-life datasets, and Theorem 4 may not work well for such datasets. Corollary 1 gives an approach for estimating the skyline cardinality over arbitrarily distributed data, in which probability functions of all dimensions are considered.

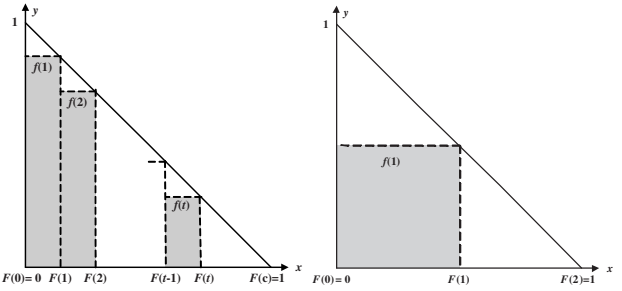
Corollary 1. *Suppose that $\xi_1\xi_2\cdots\xi_n$ are n k-dimensional live elements in a sliding window, where $\xi_i = < x_{i1}, x_{i2}, \cdots, x_{ik} >$. The probability function of the data over the jth dimension is f_j; $\mathbb{P}\{x_{ij} = v_{jt}\} = f_j(t)$, $v_{j1} < v_{j2} < \cdots < v_{jc_j}$, where c_j is the value cardinality of the jth dimension. Under assumptions of statistical independence across dimensions, the expected number of the skyline elements $\Psi(n,k)$ equals*

$$n \cdot \sum_{t_1=1}^{c_1} \cdots \sum_{t_k=1}^{c_k} f_1(t_1)\cdots f_k(t_k) \left(1 - \prod_{j=1}^{k} \sum_{\theta=1}^{t_j} f_j(\theta) \right)^{n-1}.$$

Proof. The proof borrows the same ideas from the proof of Theorem 4; for space limitations, we omit the details.

Estimating the skyline cardinality using Corollary 1 has a computational complexity of $O(\prod_{j=1}^{k} c_j)$, where c_j is the value cardinality of the jth dimension; if the number of dimensions and the value cardinalities of some dimensions are large, the computational cost overhead is unacceptable. Definition 1 gives the definition of high and low value cardinality. If a dimension was defined with high value cardinality, we may consider that there are no duplicate values over the dimension, hence the probability function of the dimension needs not to be considered and we can reduce the computational cost thereafter. With well tuned threshold value ϵ, we can get good approximation of the skyline cardinality.

Definition 1 (High and Low Value Cardinality). *Suppose that the probability function over a dimension is f; the value of a randomly selected element*



(a) high value cardinality (b) low value cardinality

Fig. 1. High and low value cardinality

has a $f(t)$ probability to be v_t, $v_1 < v_2 < \cdots < v_c$, where c is the value cardinality of the dimension; if the following inequation

$$\left| \frac{1}{2} - \sum_{t=1}^{c} f(t)\,(1 - F(t)) \right| < \epsilon, \quad F(t) = \sum_{\theta=1}^{t} f(\theta)$$

holds, where ϵ is the threshold value, we say that the dimension has high value cardinality; otherwise, the dimension has low value cardinality.

Figure 1 illustrates the high and low value cardinality defined in Definition 1, where the area of the triangle is 0.5; if the area of the greyed regions approximates 0.5, we are sure that any two values over the dimension have a very small probability to be equal, hence we may consider that there are no duplicate values over the dimension. Theorem 5 gives an efficient approach for estimating the skyline cardinality over arbitrarily distributed data; since the probability functions of dimensions with high value cardinality are no longer considered, the computational complexity is significantly reduced. The cardinality threshold ϵ in Definition 1 has great influence on the performance of this approach. If ϵ has a very small value, fewer dimensions can have high value cardinality; in this case, the computational complexity will be higher, but the result will be more accurate. Otherwise, more dimensions can have high value cardinality; in this case, the computational complexity will be lower, but the result may be less accurate.

Theorem 5. *Suppose that $\xi_1\xi_2\cdots\xi_n$ are n k-dimensional live elements in a sliding window, where $\xi_i = < x_{i1},\ x_{i2}, \cdots,\ x_{ik} >$. The data over k° dimensions has low value cardinality, and the data over each other dimension has high value cardinality; without loss of generality, suppose that the data over each of the first k° dimensions has low value cardinality, and the probability function of the data over the jth dimension is f_j; $\mathbb{P}\{x_{ij} = v_{jt}\} = f_j(t)$, $v_{j1} < v_{j2} < \cdots < v_{jc_j}$, where c_j is the value cardinality of the jth dimension. If $k = k^\circ$, the expected number of the skyline elements in the sliding window $\Psi(n, k)$ equals*

$$n \cdot \sum_{t_1=1}^{c_1} \cdots \sum_{t_k=1}^{c_k} f_1(t_1)\cdots f_k(t_k) \left(1 - \prod_{j=1}^{k} \sum_{\theta=1}^{t_j} f_j(\theta)\right)^{n-1}.$$

If $k > k^\circ$, $\Psi(n,k)$ can be approximated by

$$\sum_{t_1=1}^{c_1} \cdots \sum_{t_{k^\circ}=1}^{c_{k^\circ}} f_1(t_1) \cdots f_{k^\circ}(t_{k^\circ}) \Psi_{t_1,\cdots,t_{k^\circ}}(n,k)$$

where $\Psi_{t_1,\cdots,t_{k^\circ}}(n,k)$ can be recursively characterized as

$$\Psi_{t_1,\cdots,t_{k^\circ}}(n,k) = \Psi_{t_1,\cdots,t_{k^\circ}}(n-1,k) + \frac{\Psi_{t_1,\cdots,t_{k^\circ}}(n,k-1)}{n}$$

with initial conditions

$$\Psi_{t_1,\cdots,t_{k^\circ}}(1,k) = 1 \qquad (k \geq k^\circ + 1)$$

$$\Psi_{t_1,\cdots,t_{k^\circ}}(n,k^\circ+1) = \frac{1 - \left(1 - \prod_{j=1}^{k^\circ} \sum_{\theta=1}^{t_j} f_j(\theta)\right)^n}{\prod_{j=1}^{k^\circ} \sum_{\theta=1}^{t_j} f_j(\theta)} \quad (n \geq 1).$$

Proof. The proof borrows the same ideas from the proof of Theorem 4; for space limitations, we omit the details.

3.4 Computing Skyline Cardinality

In order to utilize the theorems presented above to estimate the skyline cardinality, we have to get the information about the distribution of the data, such as the value cardinality and the value frequency. In this subsection, we discuss how we can compute the skyline cardinality online in a data stream environment.

The Spectral Bloom Filter (SBF) [7] is an extension of the standard bloom filter [3] for supporting the estimation of the value frequency and the value cardinality. The bit vector in the standard bloom filter is replaced by a counter vector in SBF. Initially, all counters are set to 0; κ hash functions $h_1, h_2, \cdots, h_\kappa$ are used to hash elements into the counters. Three strategies, i.e. Minimum Selection (MS), Minimal Increase (MI), and Recurring Minimum (RM), are used to maintain SBF. For sliding windows, in order to estimate the skyline cardinality using Theorem 5, the RM strategy is the best choice, since it supports deletions and has relatively lower error rate. In dynamic environments, the naive method for estimating the skyline cardinality is to recompute the expected skyline cardinality using Theorems 4 or 5 whenever the distribution is changed; however, the method is both space and time inefficient and is not necessary. In our work, in the case of estimating the skyline cardinality using Theorem 4, we use a threshold value γ_c to demonstrate that when the change of the cardinality of a dimension exceeds γ_c, the expected skyline cardinality should be recomputed. In the case of estimating the skyline cardinality using Theorem 5, an additional threshold value γ_f is used to demonstrate that when the change of the frequency of a value exceeds γ_f, the expected skyline cardinality should be recomputed.

Algorithm 1: Estimating_Skyline_Cardinality(k,$\mathcal{F}$, γ_c, γ_f)

```
   Input  : k: the number of dimensions
            F: the data stream
            γc: threshold of cardinality change over a dimension
            γf: threshold of frequency change of a value
 1 begin
 2     n ← 0;
 3     while the data stream F is not terminated do
 4         wait until an element ξ arrives or expires;
 5         if ξ is an arriving element then
 6             n ← n + 1;
 7             for i ← 1 to k do
 8                 find ← sbf[i].lookfor(ξ.x[i]);
 9                 if find = false then
10                     v[i].insert(ξ.x[i]); λc[i] ← λc[i] + 1;
11                 end
12                 sbf[i].insert(ξ.x[i]); λf[i].insert(ξ.x[i]);
13             end
14         else
15             n ← n − 1;
16             for i ← 1 to k do
17                 num ← sbf[i].getnum(ξ.x[i]);
18                 if num = 1 then
19                     v[i].delete(ξ.x[i]); λc[i] ← λc[i] − 1;
20                 end
21                 sbf[i].delete(ξ.x[i]); λf[i].delete(ξ.x[i]);
22             end
23         end
24         recompute ← false;
25         for i ← 1 to k do
26             if |λc[i]| > γc then
27                 recompute ← true; break;
28             end
29             if λf[i].check(γf) = true then
30                 recompute ← true; break;
31             end
32         end
33         if recompute = true then
34             card ← computeT5(n,k,sbf,v); report(card);
35             for i ← 1 to k do
36                 λc[i] ← 0; λf[i].clear();
37             end
38         end
39     end
40 end
```

Algorithm 1 shows how to estimate the skyline cardinality over sliding windows using Theorem 5; an array of SBFs maintained by the RM strategy are used to summarize the data over each dimension. Initially, the number of the live elements in the sliding window n is set to 0 (line 2); while the stream $\mathcal{F}$ is not terminated, the algorithm waits until a new element arrives or a live element expires (line 4). If a new element ξ arrives, the number of the live elements in the sliding window is increased by 1 (line 6). Then, for each dimension of the element, determine whether the dimension value is contained by the corresponding SBF (line 8); if not contained, the dimension value is inserted into the vector $v[i]$ which consists of the distinct values over the dimension and the value cardinality change over the dimension $\lambda_c[i]$ is increased by 1 (line 10). Finally, each dimension value of the new element is inserted into the corresponding

SBF and update the vector $\lambda_f[i]$ which records the frequency changes of each distinct value over the corresponding dimension. The process of processing an expired element (lines 14-23) is just the reverse process of processing an arriving element. After processing an element, the threshold values γ_c and γ_f are used to determine whether the skyline cardinality needs to be recomputed (lines 24-32); if needs to be recomputed, recompute the expected skyline cardinality using Theorem 5 and reset λ_c and λ_f (lines 33-38). computeT5(n,k,sbf,v) computes the expected skyline cardinality, where n is the number of the live elements in the sliding window, k is the number of the dimensions, v stores the distinct values over each dimension, and sbf is the array of SBFs which can be used to estimate the number of the times that a value occurs over a dimension. Given the parameters, computing the expected skyline cardinality using Theorem 5 is rather straightforward; for space limitations, we omit the details.

The algorithm for estimating the skyline cardinality using Theorem 4 is quite similar but simpler, as we only need to record the value cardinality for each dimension and consider the only threshold γ_c. But the method may suffer from non-uniform distributions, since Theorem 4 assumes that the data over each dimension is uniformly distributed, and skewed data distribution may affect the accuracy of Theorem 4.

4 An Experimental Study

In this section, we experimentally evaluate the performance of our approaches, i.e. Theorems 4 and 5, for estimating the skyline cardinality over sliding windows in the stream environment. Since Theorems 1 and 2 are theoretically equivalent and Theorem 2 can only be approximated by some scientific computing methods, we consider Theorem 1 as a competitor of our approaches. We also compare our approaches with Theorem 3 over datasets in which the value cardinality of a dimension is limited to 2.

We use 3 hash functions and 3,000 counters for the SBF of a dimension. The algorithms are implemented by the C++ programming language and run on a 2.0GHz Intel CPU with 1GB of memory. To better show the performance under different data distributions, we conduct the experiments on synthetic datasets. We test the performance over 3-dimensional and 6-dimensional datasets which contain 30,000 elements, and the data over each dimension is generated by the GNU Scientific Library [1]. The details of the datasets used in our experiments are shown in Table 1, where c is the value cardinality of a dimension. The skewed data over a dimension is generated by two distributions alternately. One half of the skewed data submits to the uniform distribution and the other half submits to the binomial distribution with parameters p and c°. The random number generator of binomial distribution returns the number of successes in c° independent trials with probability p. For "continuous distribution" in Table 1, it represents that no duplicate values appear over a certain dimension.

For each dataset, the actual skyline cardinality is evaluated by the average number of skyline elements of the sliding window; we use the average computed

Table 1. Figures and corresponding datasets

Figure	Dataset
Figure 2(a)	2 dimensions: continuous distribution; 1 dimension: uniform distribution and $c = 50$.
Figure 2(b)	2 dimensions: continuous distribution; 1 dimension: skewed distribution, $c^\circ = 50$ and $p = 0.5$.
Figure 2(c)	1 dimension: continuous distribution; 1 dimension: uniform distribution and $c = 50$; 1 dimension: the first half data satisfies $c = 100$, the other half satisfies $c = 10$, and the data over this dimension is randomly generated.
Figure 3(a)	2 dimensions: continuous distribution; 1 dimension: uniform distribution and $c = 50$.
Figure 3(b)	1 dimension: continuous distribution; 2 dimensions: uniform distribution and $c = 50$.
Figure 3(c)	2 dimensions: continuous distribution; 1 dimension: skewed distribution, $c^\circ = 50$ and $p = 0.5$.
Figure 3(d)	1 dimension: continuous distribution; 1 dimension: skewed distribution, $c^\circ = 50$ and $p = 0.3$; 1 dimension: skewed distribution, $c^\circ = 50$, and $p = 0.7$.
Figure 3(e)	5 dimensions: continuous distribution; 1 dimension: uniform distribution and $c = 2$.
Figure 3(f)	3 dimensions: continuous distribution; 2 dimensions: uniform distribution and $c = 50$; 1 dimension: uniform distribution and $c = 2$.

skyline cardinality as the result of Theorems 4 and 5. In the experimental figures, T1, T3, T4 and T5 represent the result of Theorem 1, Theorem 3, Theorem 4, and Theorem 5 respectively.

4.1 Effect of the Thresholds

In the first set of experiments, we test the effect of the threshold ϵ, i.e. the watershed of high and low value cardinality, to the performance of Theorem 5. Figure 2(a) and Figure 2(b) illustrate the skyline cardinality with respect to ϵ over two different datasets, when the sliding window size is set to 500. In Figure 2(a), when ϵ is smaller than 0.01, the result of Theorem 5 is close to the actual skyline cardinality; but when ϵ is greater than 0.015, the results of Theorem 5 are degraded and superpose the results of Theorem 1. This happens because the dimension with uniformly distributed data is judged as a dimension with low value cardinality when $\epsilon \leq 0.01$, and hence the result is a good estimation of the actual skyline cardinality. The dimension is misjudged as a dimension with high value cardinality when $\epsilon \geq 0.015$. Thus the three dimensions are all with high value cardinality. Theorem 5 is equivalent to Theorem 1 at this moment and the results prove this point. The watershed of high and low value cardinality in Figure 2(b) is greater than that in Figure 2(a), because the value of $|1/2 - \sum_{t=1}^{c} f(t)\,(1 - F(t))\,|$ (see Definition 1) of a dimension in Figure 2(b) is greater than that in Figure 2(a) due to the skewed data distribution. From above results, we can see that the selection of ϵ is important for better performance of Theorem 5, and we fix the cardinality threshold ϵ at 0.005 in the following experiments.

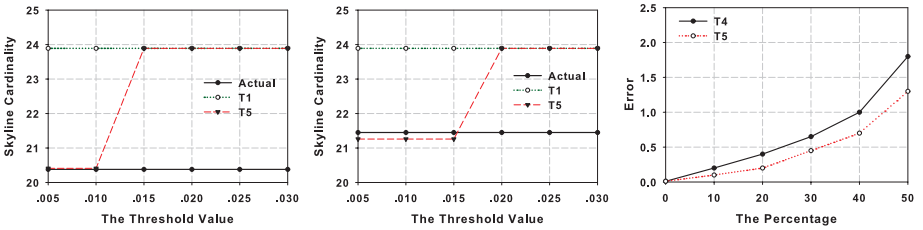


Fig. 2. The effect of thresholds

Next, we evaluate the effect of the recomputation threshold on the accuracy of skyline cardinality estimation. We fix the window size at 500. γ_c which is the threshold of cardinality change over a dimension and γ_f which is the threshold of frequency change of a value are given the same value for ease of the presentation. We only need to examine γ_c for Theorem 4, while both γ_c and γ_f for Theorem 5. In this experiment, we vary the threshold from 0% to 50% and calculate the average error and times of recomputation. For example, 10% means that we do not recompute the skyline until the change of value cardinality (value frequency) on any dimension is larger than 10%. The average error stands for the difference between the results of Theorems 4(5) and results of respective Theorems with the certain percentage of changes. As shown in Figure 2(c), both the errors of Theorem 4 and Theorem 5 increase when $\gamma_c(\gamma_f)$ increases. However, the error is not significant compared with skyline cardinality which is around 20, as we recompute the skyline once the change on any dimension exceeds the threshold. We also find that the error of Theorem 5 is smaller than that of Theorem 4. The reason is that Theorem 4 computes the skyline cardinality only considering the change of value cardinality, while Theorem 5 may recompute when the data distribution changes.

4.2 Performance over Different Datasets

Figure 3(a) to Figure 3(f) present the experimental results of different approaches under various data distributions. The actual skyline cardinality and the estimated skyline cardinality of different methods increase when the window size increases, as more objects need to be evaluated.

For the given window size and the number of dimensions, the results of Theorem 1 remain the same regardless of the change of data distributions. Therefore, Theorem 1 shows a poorer performance for the datasets, in which not all the dimensions are of continuously distributed data. In Figure 3(a) and Figure 3(b), both Theorem 4 and Theorem 5 provide a good estimation for the actual skyline cardinality and this fully supports the effectiveness of our methods. One dimension with continuously distributed data is replaced by one dimension with uniformly distributed data in Figure 3(b), and the actual skyline cardinality decreases accordingly for the intuitionistic reason that when the value cardinality

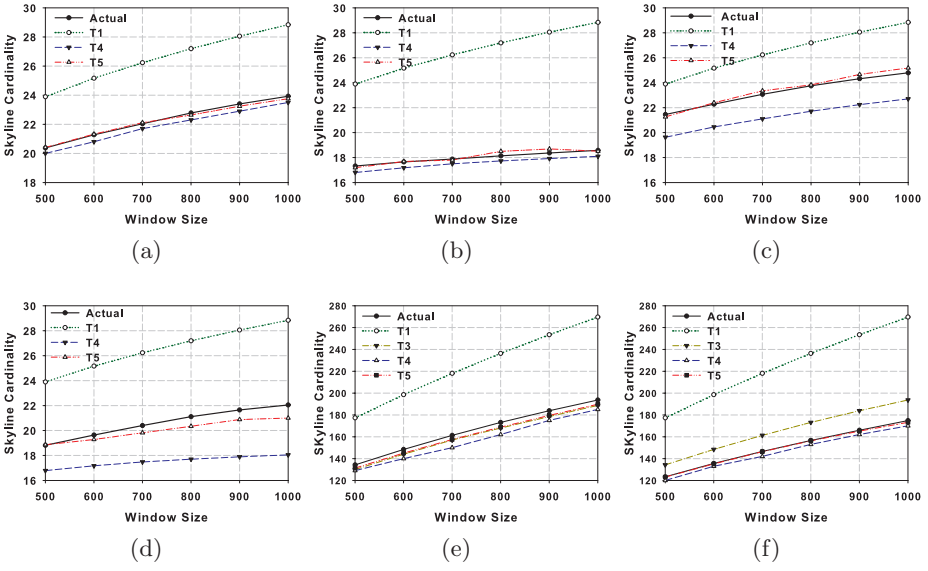


Fig. 3. Performance under different datasets

of a dimension is reduced, more elements are probably dominated. The skewed data distribution is introduced in Figure 3(c) and Figure 3(d), and Theorem 4 performs worse than Theorem 5 as we expect. The actual skyline cardinality in Figure 3(d) is smaller than that in Figure 3(c) because of the reduced value cardinality of one dimension. Comparing Figure 3(a) and Figure 3(c), we can find that the live elements in the window have higher probabilities to be skyline elements under skewed data distribution.

Next, we consider Theorem 3 as a competitor. The dataset used in Figure 3(e) satisfies the assumptions of Theorem 3. We can see that the result of Theorem 3 is almost as good as that of Theorem 5. While in Figure 3(f), we cannot compute the results of Theorem 3 directly as two dimensions are not continuously distributed. We treat them as continuously distributed to approximate the result for Theorem 3. Both Theorem 4 and Theorem 5 outperform Theorem 3, because the preconditions of Theorem 3 does not consider the data distribution whose value cardinality is greater than 2.

5 Conclusion

In this paper, we relaxed the strong assumptions of previous work and proposed Theorem 4, which only uses the value cardinality of each dimension to estimate the skyline cardinality under the assumption that the data over each dimension is uniformly distributed. We also gave a robust approach, i.e. Theorem 5, to effectively estimate skyline cardinality over arbitrarily distributed data. To apply

Theorem 4 and Theorem 5 in the data stream environment, we used SBF to capture the value cardinality and the probability that a value occurs over a dimension. Finally, we compared our approaches with all previous approaches by extensive experimental study, and our approaches, especially Theorem 5, significantly outperform previous approaches.

References

1. http://www.gnu.org/software/gsl/
2. Bentley, J.L., Kung, H.T., Schkolnick, M., Thompson, C.D.: On the average number of maxima in a set of vectors and applications. J. ACM 25(4), 536–543 (1978)
3. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. Communications of the ACM 13(7), 422–426 (1970)
4. Börzsönyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: Proceedings of ICDE 2001, pp. 421–430 (2001)
5. Buchta, C.: On the average number of maxima in a set of vectors. Inf. Process. Lett. 33(2), 63–65 (1989)
6. Chaudhuri, S., Dalvi, N.N., Kaushik, R.: Robust cardinality and cost estimation for skyline operator. In: Proceedings of ICDE 2006, p. 64 (2006)
7. Cohen, S., Matias, Y.: Spectral bloom filters. In: Proceedings of SIGMOD 2003, pp. 241–252 (2003)
8. Godfrey, P.: Skyline cardinality for relational processing. In: Seipel, D., Turull-Torres, J.M.a. (eds.) FoIKS 2004. LNCS, vol. 2942, pp. 78–97. Springer, Heidelberg (2004)
9. Kossmann, D., Ramsak, F., Rost, S.: Shooting stars in the sky: An online algorithm for skyline queries. In: Proceedings of VLDB 2002, pp. 275–286 (2002)
10. Kung, H.T., Luccio, F., Preparata, F.P.: On finding the maxima of a set of vectors. J. ACM 22(4), 469–476 (1975)
11. Lin, X., Yuan, Y., Wang, W., Lu, H.: Stabbing the sky: Efficient skyline computation over sliding windows. In: Proceedings of ICDE 2005, pp. 502–513 (2005)
12. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: Proceedings of SIGMOD 2003, pp. 467–478 (2003)
13. Rosen, K.H.: Discrete Mathematics and Its Applications, 4th edn. WCB/McGraw-Hill, Boston (1999)
14. Tao, Y., Papadias, D.: Maintaining sliding window skylines on data streams. IEEE Trans. Knowl. Data Eng. 18(2), 377–391 (2006)