Unrestricted Wavelet Synopses under Maximum Error Bound

Chaoyi Pang Qing Zhang David Hansen Anthony Maeder
The Australian E-Health Research Centre, CSIRO, Australia
firstname.secondname@csiro.au

ABSTRACT

Constructing Haar wavelet synopses under a given approximation error has many real world applications. In this paper, we take a novel approach towards constructing unrestricted Haar wavelet synopses under an error bound on uniform norm (L_∞). We provide two approximation algorithms which both have linear time complexity and a $(\log N)$-approximation ratio. The space complexities of these two algorithms are $O(\log N)$ and $O(N)$ respectively. These two algorithms have the advantage of being both simple in structure and naturally adaptable for stream data processing. Unlike traditional approaches for synopses construction that rely heavily on examining wavelet coefficients and their summations, the proposed construction methods solely depend on examining the original data and are extendable to other findings. Extensive experiments indicate that these techniques are highly practical and surpass related ones in both efficiency and effectiveness.

1. INTRODUCTION

Widely used in signal and image processing, the wavelet technique has been considered very promising for data compression and query approximation in database domains [11, 16]. Recent research has applied wavelet synopses to summarize data streams for approximate queries [6, 9]. The basic idea of constructing a wavelet synopsis of a data vector, with size N, is to first transform the data vector into a representation with respect to a wavelet basis. Then the data vector is approximated by retaining M coefficients as the wavelet synopsis and setting those remaining to 0 implicitly.

A conventional approach is to find M coefficients to minimize the overall mean squared error (L_2) [15]. This can be solved gracefully by applying the Parseval's theorem [2]. However, the main drawback of this approach is that users cannot control the approximation error of individual elements in the data vector. This severely impedes further applications of the wavelet approximation. To alleviate this, researchers have made efforts to construct wavelet synopses with guaranteed maximum error in approximation under L_∞ metric [2]. Two approaches for constructing wavelet synopses under uniform norm L_∞ have been taken: one is to construct bucket bound (size bound) synopses which would minimize the maximum approximation error of single data elements [3] whilst the other is to construct the smallest size of synopses such that the maximum approximation error does not exceed a given error bound on L_∞ metric [12, 14]. The details are explained below.

Bucket bound synopses: The goal is to construct a synopsis of at most B Haar wavelet coefficients to minimize the approximation error under L_∞ metric. The bucket bound (size-bound) synopses have been studied intensively [2, 3, 4, 8, 9, 10]. The construction of synopses has $O(N^2)$ time/space complexity [3]. Several methods have been proposed to improve the performance [4, 6, 9]. Guha et al [5, 6] indicate that the restriction to Haar wavelet coefficients for the elements of a synopsis is not a good strategy in approximation and extend to the construction of synopses not restricted to Haar wavelet coefficients. Let Δ be an approximation error bound on L_∞ metric. They tackle the problem of building unrestricted bucket bound synopses in time $O((\frac{\Delta}{\delta})^2 N \log N \log^2 B)$ through using rounding techniques and resolution parameter δ on error bound Δ. Karras et al [10] improve the above result close to $O((\frac{\Delta}{\delta})^2 N \log N)$.

Nevertheless, many real applications suggest the construction of error bound (Δ-bound) synopses as stated below.

Restricted error-bound synopses: Given an error bound Δ on uniform norm L_∞, the goal is to find M_{op}, a Haar wavelet synopsis with the smallest set of coefficients among all possible solutions, that would satisfy the Δ bound on L_∞ (i.e., $L_\infty < \Delta$) in approximation. M_{op} is called *R-optimal* as each element in the synopsis is restricted to be a Haar coefficient.

The error bound synopses have been studied by Muthukrishnan and Guha [4, 12]. Their approaches for this problem are basically similar to those for bucket bound synopses. The construction of the optimal synopsis M_{op} has $O(N^2)$ time complexity. Furthermore, the construction of "unrestricted" error-bound synopses are yet to be fully studied.

Unrestricted error-bound synopses: Given error bound Δ, the goal is to find S_{op}, a synopsis with the smallest set of unrestricted coefficients among all possible solutions, that would satisfy the Δ bound on L_∞. S_{op} is called *U-optimal* as each element in the synopsis is not restricted to be a Haar coefficient.

Fortunately, extended from the idea of [5, 6] on unrestricted bucket bound synopses, Karras et al propose an efficient algorithm[1] for the U-optimal problem under two parameters δ and Δ. This most recent result (KSM algorithm) uses the dynamic-programming framework and has $O((\frac{\Delta}{\delta})^2 N)$ time complexity. Theoretically, their algorithm can obtain an U-optimal solution through (iteratively) using smaller δ values. The execution time of their algo-

[1]The MinHaarSpace algorithm of [10]. It is termed as KSM algorithm in this paper.

EDBT 2009, March 24–26, 2009, Saint Petersburg, Russia.

rithm greatly depends on the value of δ and Δ: It derives a smaller sized synopsis under a smaller δ value but requires a greater length of time[2].

Given a data vector and an error bound Δ, the objective of this paper is to provide efficient algorithms for constructing an unrestricted synopsis S such that the maximum approximation error of each single data element is bounded by Δ and S is quite close to S_{op} in size.

Motivations: The unrestricted error-bound synopses are preferred in many applications where the size-bound synopses can be naturally unadaptable. For example, the correlation analysis on the surgery patients requires the physiological data collected during operations to be stored in the database in sampling manner for the storage and query concerns. The size-bound synopses would not be suitable for this scenario due to the variations in the durations of surgery operations and may generate unqualified samplings. Furthermore, the major issues in this application are time efficiency and compression quality: The restriction of Haar wavelet coefficients samples has little effect.

Examples of physiological data include electrocardiogram, arterial blood pressure, central venous pressure and respiration etc.

Contributions: We provide two linear-time algorithms on constructing the unrestricted error bound synopses: Fixed-value Shift (F-Shift) algorithm and Sliding-value Shift (S-Shift) algorithm. F-Shift is an approximation algorithm for the unrestricted absolute error-bound problem while S-Shift can be viewed as an optimal version of the former. That is, S-Shift always generates smaller (or equal) sized synopses than that of F-Shift. Compared with conventional approaches, our contributions in this paper can be summarized as follows:

(1) We provide two computationally efficient and effective approximation algorithms to construct unrestricted error-bound synopses under L_∞ metric. Let N be the size of the data vector. Our two algorithms, F-Shift and S-Shift, both have linear time complexity. F-Shift has $O(\log N)$ space complexity and S-Shift has $O(N)$ space complexity. They are $(\log N)$-approximation algorithms. Let S_F and S_S represent the synopses constructed by our F-Shift and S-Shift algorithms respectively. Then $|S_S| \leq |S_F| \leq |S_{op}| \log N$ holds for the optimal synopsis S_{op}. Furthermore, we show that this property does not hold for R-optimal synopsis M_{op}. We indicate the size of M_{op} can be very large even if S_{op} is very small (Example 4.6). This result suggests that unrestricted synopses such as S_F or S_S can be preferable to restricted synopses in real applications.

(2) Our approach in constructing unrestricted synopses is novel. The traditional methods usually work on the decomposed data (coefficients) and find the retained coefficients either by checking the combinatorial path summations [4, 6, 10, 12] or by retaining the most "significant" coefficients [9]. Instead, we construct synopses by "shifting" (shrinking) the adjacent data ranges, which will be explained later in detail.

(3) The shift algorithms are directed by and based on the three novel concepts along with their relevant properties proposed in the paper: shift transformation, data scope and shift range. These properties lead to the correctness of the Shift algorithms and the above-listed features. The same methodology can be extended to other findings besides the construction of synopses on relative errors [13].

(4) The efficiencies and effectiveness of Shift algorithms have been demonstrated through extensive experiments on both synthetic data and real life data. In terms of effectiveness, the Shift algorithms generate error-bound synopses with good compression quality. Although the approximation ratio on synopses size is $\log N$ in theory, the practical tests indicate that $|S_S|$ is often quite close to the size of optimal synopses. In terms of efficiencies, the Shift algorithms dramatically improve the synopses construction time against the KSM algorithm under a same compression quality goal.

(5) Moreover, due to $O(\log N)$ space complexity, the F-Shift algorithm can be used as an online compression algorithm for stream data: the algorithm processes incoming data progressively and does not require the Haar wavelet error tree to be pre-computed. In the process of compressing stream data, algorithms with linear (or above) space complexity cannot be used directly and require a mechanism to segment incoming data into sections (e.g., fixed windows). In contrast, unlike those existing compression methods including those on L_p $(1 \leq p < \infty)$ measure, the F-Shift algorithm can compress stream data directly in a synchronized way.

The rest of the paper is organized as follows. Section 2 explains the basic terminologies. Section 3 describes the shift transformation and defines the data scope and the shift range concepts, in conjunction with their related properties. Section 4 introduces the F-Shift algorithm and its properties. Section 5 is about the S-Shift algorithm. Section 6 reports our experiment results. Section 7 concludes this paper.

Symbol $i \in \{0..N-1\}$	**Description**
$D, [d_0; ..; d_{N-1}]$	data vector
W_D	Haar wavelet transformation on D
T	error tree
c_i, s_i	coefficient node, shift coefficient node
d_i, $\hat{d}_i$	leaf/data node and its reconstruction
$path(u)$	all ancestors of node u in T
$T(c)$	subtree rooted at c
$T_L(c)/T_R(c)$	left/right subtree of $T(c)$
Δ	error bound on approximation (> 0)
S	set of shift coefficients
S_F/S_S	synopsis obtained by F-Shift/S-Shift
M	set of Haar wavelet coefficients
M_{op}/S_{op}	restricted/unrestricted optimal synopsis
$T\langle S\rangle$	error tree after S shift transform on T

Table 1: Notations

2. HAAR WAVELET AND DATA APPROXIMATION

Table 1 summarizes the notations used throughout this paper.

Wavelets are a mathematical tool for hierarchically decomposing functions. Generally, the wavelet decomposition of a function consists of a coarse overall approximation together with detail coefficients that influence the function at various scales [15]. It can generally be computed in linear time.

The Haar wavelet is the simplest wavelet. The Haar wavelet decomposition of a data vector consists of a coefficient representing the overall average of the data values followed by detail coefficients in the order of increasing resolution. Each detail coefficient is the difference of a pair of averaged values from the lower level.

Suppose we are given data vector $D = [19; 17; 12; -4; 7; -1; -3; -7]$ containing $N = 8$ data values. The Haar wavelet decomposition of D can be computed as follows. We first average the data values, pairwise, to get a new lower-resolution data with values $[18; 4; 3; -5]$. That is, the first two values in the original data (19 and 17) average to 18, and the second two values 12 and -4

[2]As stated in [10]: Smaller (δ) values burdened the running time without significant quality increase; larger (δ) values were undermining the quality of the synopses.

average to 4, and so on. We also store the pairwise differences of the original values (divided by 2) as detail coefficients: $[\frac{19-17}{2}; \frac{12+4}{2}; \frac{7+1}{2}; \frac{-3+7}{2}]$, which is $[1; 8; 4; 2]$. It is easy to see that the original values can be recovered from the averages and differences. By repeating this process recursively on the averages, we get the Haar wavelet decomposition in Figure 1(i). We define the wavelet decomposition W_D of data vector D to be the coefficient representing the overall average of the original data, followed by the detail coefficients in the order of increasing resolution. That is, $W_D = [5; 6; 7; 4; 1; 8; 4; 2]$.

The Haar wavelet decomposition can be expressed in an error tree structure [11]. Depicted in Figure 1(ii), error tree T is a hierarchical structure representing the nodes of W_D and D, where each internal node c_i represents a wavelet coefficient and each leaf node d_i represents an original data item. We use $T(u)$ to denote the subtree rooted at u, $T_L(u)$ and $T_R(u)$ as the subtrees rooted at the left and right child of u respectively.

Given a node u (internal or leaf), we define $path(u)$ as the set of nodes that lie on the path from the root node to u (excluding u). To reconstruct any leaf node d_i through the error tree T, we only need to compute the signed summation of nodes belonging to $path(d_i)$. That is,

$$d_i = \sum\nolimits_{c_j \in path(d_i)} \delta_{ij} c_j, \quad (1)$$

where $\delta_{ij} = +1$ if $d_i \in T_L(c_j)$ and $\delta_{ij} = -1$ if $d_i \in T_R(c_j)$. From this property, we define an error tree to be a binary tree where each leaf node d_i equals the signed summation of all the internal nodes of $path(d_i)$. That is, each d_i satisfies Equation (1).

Being reconstructed from W_D, data vector D can also be approximated with a subset of W_D. Let $\hat{d}_i$ denote the approximated data value of d_i on $M \subseteq W_D$. That is, $\hat{d}_i = \sum_{c_j \in \mathit{path}(d_i) \cap M} \delta_{ij} c_j$. Rather than obtaining the minimum set $M \subseteq W_D$ that satisfies $\max\limits_{0 \le i < N} |\hat{d}_i - d_i| < \Delta$ for a given error bound $\Delta > 0$, in this paper, we will investigate how to find a close-to-minimum-sized S such that $\max\limits_{0 \le i < N} |\hat{d}_i^{(S)} - d_i| < \Delta$ where

$$\hat{d}_i^{(S)} = \sum_{c_j \in \mathit{path}(d_i) \wedge s_j \in S} \delta_{ij} s_j. \quad (2)$$

Generally, S is a set of s_i obtained from a subset of W_D by representing each c_j in the subset with s_j. Equation (2) is the signed summation of S nodes on *path*(d_i). Detailed discussion on S and its graphic meaning will be given in the next section.

EXAMPLE 2.1. *In error tree T of Figure 1(ii), $T(c_5) = T_R(c_2)$ and $T(c_4) = T_L(c_2)$. Data value d_2 can be reconstructed through the nodes of $path(d_2)$, i.e., $d_2 = 5 + 6 + (-7) + 8$. Let $M = \{c_1, c_2, c_3, c_5, c_6\}$ and $S = \{s_0, s_1, s_5\}$ where[3] $s_0 = 5.5$, $s_1 = 5.75$ and $s_5 = 8$. Then $\hat{d}_2 = 6 + (-7) + 8$, which is 7, and $\hat{d}_2^{(S)} = 5.5 + 5.75 + 8$, which is 19.25.*

3. CONCEPTS AND PROPERTIES

In this section, three major concepts of this paper: *shift transformations*, *data scope* and *shift range* are defined. Some properties for each concept are given. We will firstly show how to use shift transformations to generate synopses. We then indicate that data scopes and shift ranges can be used to select shift coefficients and for assigning shift values. With these results, the general idea of Shift algorithms is given in Section 3.4.

[3]We will explain why we chose these particular values for s_i in Section 3.3.

3.1 Shift Transformation

Shift transformations supply graphic explanations and meanings for the unrestricted synopses and their formations. In this subsection, we first define shift transformations. We then show that a synopsis is a set of shift values generated from shift transformations that transform the error tree to "Δ-bound" (Property 3.3).

Let c_i be a coefficient of error tree T and s_i be a real value. An *s_i-shift transformation* at c_i of T transforms T into $T' = T\langle s_i \rangle$ through the following steps:

1. Replace coefficient c_i with $c_i - s_i$;
2. When $i > 0$,
   (a) replace each leaf node d of $T_L(c_i)$ with $d - s_i$;
   (b) replace each leaf node d of $T_R(c_i)$ with $d + s_i$.
3. When $i = 0$, replace each leaf node d of T with $d - s_0$.

s_i is called the *shift coefficient* (at c_i). In general, suppose that S is a set of shift coefficients on error tree T. $T\langle S \rangle$ is the tree transformed from T through applying the above steps on each $s_i \in S$ iteratively. Clearly, $T\langle S \rangle$ is well defined and irrelevant to the transformation order of the elements in S.

From the error tree definition, it can be proven that $T\langle S \rangle$ is the error tree on the updated data vector.

PROPERTY 3.1. *Let T be an error tree and S be a set of shift coefficients on T. Then $T\langle S \rangle$ is also an error tree.*

The above property assures that the Shift algorithms introduced in the paper work on an error tree at each iteration.

EXAMPLE 3.2. *(Continued from Example 2.1) Error trees of $T\langle \{s_5\} \rangle$, $T\langle \{s_1, s_5\} \rangle$ and $T\langle \{s_0, s_1, s_5\} \rangle$ are expressed from Figure 1(iii) to Figure 1(v) respectively.*

A set of shift coefficients S is called a Δ-bound unrestricted synopsis (or simply *synopsis*) if $|\mathit{diff}_S(d_i)| < \Delta$ holds for each d_i of T where $\mathit{diff}_S(d_i) = d_i - \hat{d}_i^{(S)}$.

Generally, a shift coefficient s_i of S may not have the same value as c_i. A *restricted synopsis* is a synopsis where each $s_i = c_i$ in S. In this paper, a restricted synopsis will be denoted by M. An *optimal* (restricted/unrestricted) synopsis is a synopsis that has the smallest size among all possible synopses. By M_{op} (and S_{op}) we denote the optimal restricted (and unrestricted) synopsis respectively. Clearly, $|M_{op}| \le |M|$ and $|S_{op}| \le |S|$ hold.

Error tree T is *Δ-bound* if $|d_i| < \Delta$ holds for each d_i of T. The following results can be directly proven from the definition of synopses. It assures the correctness of Shift algorithms and indicates that a synopsis can be derived from shift transformations.

PROPERTY 3.3. *Let T be an error tree.*

1. *$T\langle S_{op} \rangle$ is Δ-bound.*
2. *Empty set $\emptyset$ is a synopsis iff T is Δ-bound.*
3. *S is a synopsis of T iff $T\langle S \rangle$ is Δ-bound.*
4. *If T is Δ-bound, then $|c_i| < \Delta$ holds for each $c_i \in T$.*

Property 3.3(4) can be proven by borrowing Property 1 of [14]. Roughly, if each $|d_i| < \Delta$, then the averages of any data values are less than Δ. Thus the averages of their differences are less than Δ. That is, $|c_i| < \Delta$ for each $c_i \in T$.

EXAMPLE 3.4. *(Continued from Example 3.2) Let $\Delta = 8$. It can be verified that $M_{op} = \{c_1, c_2, c_3, c_5, c_6\}$ and $S_{op} = \{s_0, s_1, s_5\}$ hold. $M = \{c_0, c_1, c_2, c_3, c_5, c_7\}$ is a restricted synopsis. Figure 1(v) is $T\langle S_{op} \rangle$ and Figure 1(vi) is $T\langle M_{op} \rangle$. Clearly, $|d_i| < \Delta$ holds for each $d_i \in T\langle S_{op} \rangle$ (or $d_i \in T\langle M_{op} \rangle$).*

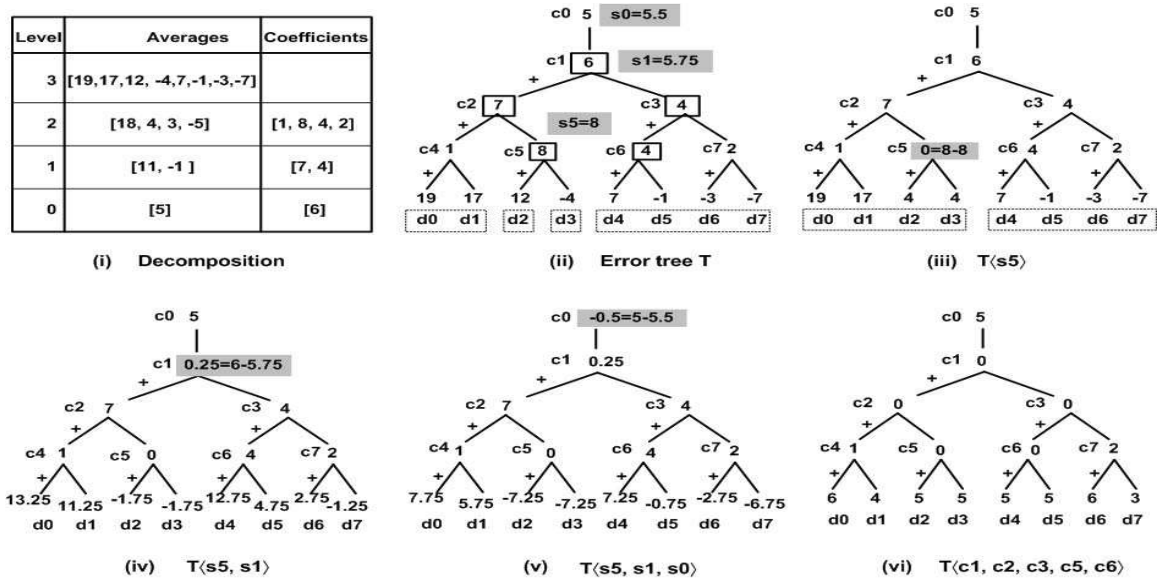


Figure 1: Example ($\Delta = 8$): $\mathbf{M_{op}} = \{\mathbf{c_1}, \mathbf{c_2}, \mathbf{c_3}, \mathbf{c_5}, \mathbf{c_6}\}$ and $\mathbf{S_{op}} = \{\mathbf{s_0}, \mathbf{s_1}, \mathbf{s_5}\}$.

3.2 Data Scope

Property 3.3 shows that a synopsis can be obtained through shift transformations. One of the remaining questions is which coefficient needs to be shifted or which s_i should be in a synopsis? In order to answer this question, we propose the *data scope* concept.

DEFINITION 3.5. *Let T' be a subtree of error tree T and S be a set of shift coefficients. The (data) scope of T' on S is defined as*

$$scope(T', S) = \frac{1}{2} \max_{d_i, d_j \in T'} |diff_S(d_i) - diff_S(d_j)| .$$

Intuitively, $scope(T', S)$ describes the data extents of $T'\langle S\rangle$ and measures $\max_{d'_i, d'_j \in T'\langle S\rangle} |d'_i - d'_j|$. With this concept, we will explain when a coefficient of a subtree needs to be shifted.

Let S be a set of shift coefficients. For simple expressions, we denote $\{s_i | c_i \in T(c) \wedge s_i \in S\}$, the subset of S that locates in $T(c)$, by $T(c) \sqcap S$. The following property suggests that the exhaustive combinations on $path(d_i)$ as in [12, 4] can be alleviated by checking data scopes.

PROPERTY 3.6. *Let S be a set of shift coefficients. Then, for any coefficient c of T,*

(i) *$scope(T(c), M_{op}) < \Delta$ and $scope(T(c), S_{op}) < \Delta$ hold;*

(ii) *$scope(T(c), S) = scope(T(c), T(c) \sqcap S)$;*

(iii) *if $scope(T(c), \emptyset) \geq \Delta$, then $T(c) \cap M_{op} \neq \emptyset$;*

(iv) *if $scope(T(c), \emptyset) \geq \Delta$, then $T(c) \sqcap S' \neq \emptyset$ holds for any synopsis S'.*

PROOF. The proof of (i) can be obtained from the definitions of M_{op}, S_{op} and data scope directly.

The proof of (ii): Let d_i and d_j be any two data nodes in $T(c)$. $\hat{d}_i^{(S)}$ and $\hat{d}_j^{(S)}$, the reconstructed d_i and d_j on S, are built from the summation of two parts: (a) $\hat{d}_{i1}^{(S)}$ and $\hat{d}_{j1}^{(S)}$ from the nodes of $(T - T(c)) \sqcap S$ where $\hat{d}_{i1}^{(S)} = \hat{d}_{j1}^{(S)}$ and, (b) $\hat{d}_{i2}^{(S)}$ and $\hat{d}_{j2}^{(S)}$; where $\hat{d}_{i2}^{(S)}$ and $\hat{d}_{j2}^{(S)}$ from the nodes of $T(c) \sqcap S$. Since $\hat{d}_h^{(S)} = \hat{d}_{h1}^{(S)} + \hat{d}_{h2}^{(S)}$ and $diff_S(d_h) = d_h - \hat{d}_h^{(S)}$ for $h = i, j$,

$$\begin{aligned} diff_S(d_i) - diff_S(d_j) &= (d_i - \hat{d}_i^{(S)}) - (d_j - \hat{d}_j^{(S)}) \\ &= (d_i - \hat{d}_{i1}^{(S)}) - (d_j - \hat{d}_{j1}^{(S)}). \end{aligned}$$

That is, $|diff_S(d_i) - diff_S(d_j)| = |diff_{T(c)\sqcap S}(d_i) - diff_{T(c)\sqcap S}(d_j)|$. As d_i and d_j are arbitrary data of $T(c)$, (ii) is proven from the definition of data scope.

As the proofs of (iii) and (iv) are similar, we will only prove (iii) in the following.

Otherwise, assume that $T(c) \cap M_{op} = \emptyset$ holds. From (ii), we have $scope(T(c), M_{op}) = scope(T(c), \emptyset)$. Since $scope(T(c), \emptyset) \geq \Delta$, $scope(T(c), M_{op}) \geq \Delta$ holds. Therefore, (iii) is proven, as it is contradictory to (i). □

We will call subtree $T(c)$ *scope bound* (or simply, *s-bound*) if $scope(T(c), \emptyset) < \Delta$, otherwise, it is *s-unbound*. $T(c)$ is called $\overline{s}$-bound (i.e., maximal s-bound) if it is s-bound and $T(c')$ is s-unbound where c' is the parent node of c in error tree T. Similarly, denote $T(c)$ to be $\underline{s}$-unbound if $T(c)$ is s-unbound but each of its subtrees is s-bound. Clearly, if $T(c)$ is Δ-bound then it is s-bound. These concepts will be used in the Shift algorithms and for the size estimation on obtained synopses.

Property 3.6(ii) says that $scope(T(c), S)$ is unchangeable by shifting any nodes of $T - T(c)$. That is, shifting the coefficients that locate outside of $T(c)$ will not change the value of $scope(T(c), S)$. Property 3.6(iii) and (iv) imply that, in order to obtain a synopsis for the given Δ, some nodes inside $T(c)$ must be shifted whenever $T(c)$ is $\underline{s}$-unbound. This is the basic property for the Shift algorithms.

EXAMPLE 3.7. *(Continued from Example 3.4) For error tree T of Figure 1(ii), $T(c_4)$, $T_L(c_5)$, $T_R(c_5)$ and $T(c_3)$ are $\overline{s}$-bound. $T(c_4)$ is also Δ-bound. The number of $\overline{s}$-bound subtrees of T is 4. Alternatively, $T(c_5)$ is $\underline{s}$-unbound. $T(c_2)$ is s-unbound as $scope(T(c_2), \emptyset) = (19 + 4)/2$ is greater than Δ. According to Property 3.6, at least one coefficient of $T(c_2)$ and $T(c_5)$ need to be shifted for a synopsis.*

3.3 Shift Range

To obtain a synopsis, Property 3.6 indicates that a shift coefficient in a $\underline{s}$-unbound subtree should be retained or shifted. It is still not clear which specific coefficient should be chosen and in what value, considering that the shift coefficient s_i will usually not have the same value as coefficient c_i. These questions will be answered in this subsection. Explicitly, we will show how to convert an $\underline{s}$-unbound subtree into an s-bound subtree by applying a shift transformation on its root node. We then give the definition of the shift range. Generally, the shift range on coefficient c_i is a range of values for s_i to choose from.

In the following, Property 3.8 actually points out a method of converting an arbitrary error tree into a Δ-bound tree through shift transformations. Significantly, it shows that only node c_i where $T(c_i)$ is $\underline{s}$-unbound and/or node c_0 need to be shifted in the process of generating a synopsis from bottom-up.

PROPERTY 3.8. *Let $T' = T(c_i)$ be a subtree of error tree T rooted at node c_i ($i > 0$) and $\Delta > 0$ be a given error bound.*

i. *If T' is $\underline{s}$-unbound for Δ, then there exists s_i such that $T'\langle s_i\rangle$ is s-bound.*

ii. *If error tree T is s-bound but not Δ-bound, then there exists s_0 such that $T\langle s_0\rangle$ is Δ-bound.*

PROOF. The proof of (**i**). Let d_{L_g} and d_{L_s} (or d_{R_g} and d_{R_s}) be the largest and smallest data values of $T_L(c_i)$ (or $T_R(c_i)$) respectively. Since T' is $\underline{s}$-unbound, the following Equations hold.

$$\begin{cases} d_{L_g} - d_{L_s} < 2\Delta, \quad d_{R_g} - d_{R_s} < 2\Delta, \\ \max\{d_{L_g}, d_{R_g}\} - \min\{d_{L_s}, d_{R_s}\} \geq 2\Delta. \end{cases} \tag{3}$$

In order to convert T' into an s-bound one, the shift coefficient s_i should satisfy $-2\Delta < d_{L_g} - s_i - (d_{R_s} + s_i) < 2\Delta$ and $-2\Delta < d_{R_g} + s_i - (d_{L_s} - s_i) < 2\Delta$. Since $d_{L_g} - d_{R_s} \geq -(d_{R_g} - d_{L_s})$, this equation can be simplified into:

$$(d_{L_g} - d_{R_s})/2 - \Delta < s_i < -(d_{R_g} - d_{L_s})/2 + \Delta. \tag{4}$$

By Equation (3), since

$$\begin{aligned} &[-(d_{R_g} - d_{L_s})/2 + \Delta] - [(d_{L_g} - d_{R_s})/2 - \Delta] \\ &= \quad 2\Delta - [(d_{R_g} - d_{R_s}) + (d_{L_g} - d_{L_s})]/2 \\ &> \quad 0, \end{aligned}$$

the existence of s_i is proven.

The proof of (**ii**). Suppose that d_g and d_s are the largest and smallest data values of $T(c_0)$ respectively. Let $x_1 = (d_g + d_s)/2$ and $l_1 = (d_g - d_s)/2$. To convert T into a Δ-bound one, s_0 should satisfy $-\Delta < x_1 - l_1 - s_0$ and $x_1 + l_1 - s_0 < \Delta$, or equivalently,

$$|x_1 - s_0| < \Delta - l_1. \tag{5}$$

From the assumption that T is s-bound, $\Delta - l_1 > 0$ holds. Therefore, Equation (5) has solutions and $T\langle s_0\rangle$ is Δ-bound for each solution s_0 of Equation (5). □

It should be noted that $T'\langle s_i\rangle$ is not guaranteed to be s-bound if we choose s_i to be c_i in Property 3.8(i). Similarly, in Property 3.8(ii), $T\langle s_0\rangle$ may not be guaranteed to be Δ-bound by setting s_0 to be c_0. That is, c_i (or c_0) may not satisfy Equation (4) (or Equation (5)).

The above proof is constructive. The proof also illustrates what values the shift coefficient needs to be chosen from. For instance, in the proof of Property 3.8(**i**), $T'\langle s_i\rangle$ is s-bound for any s_i that is in the range of Equation (4).

Furthermore, a special situation of Equation (4) is to minimize the scope of $T'\langle s_i\rangle$: in addition to being s-bound, shift the two intervals of $(d_{L_s} - s_i, d_{L_g} - s_i)$ and $(d_{R_s} + s_i, d_{R_g} + s_i)$ enclosed, i.e., one interval subsumes another. In this situation, it can be verified that s_i satisfies the following equation:

$$\begin{cases} \min\{(d_{L_g} - d_{R_g})/2,\ (d_{L_s} - d_{R_s})/2\} \leq s_i \\ s_i \leq \max\{(d_{L_g} - d_{R_g})/2,\ (d_{L_s} - d_{R_s})/2\}. \end{cases} \tag{6}$$

Let $l_L = (d_{L_g} - d_{L_s})/2$, $l_R = (d_{R_g} - d_{R_s})/2$, $x_L = (d_{L_g} + d_{L_s})/2$ and $x_R = (d_{R_g} + d_{R_s})/2$. Equation (6) can be rewritten into

$$|(x_L - x_R) - 2s_i| \leq |l_L - l_R|. \tag{7}$$

The existence of s_i in Equation (7) can be proven from Equation (6). We will use Equation (7) to select the values for s_i whenever $T(c_i)$ is $\underline{s}$-unbound rather than through Equation (4). In Section 5.1, we will explain why we do not use Equation (4) in our approach.

DEFINITION 3.9. *Let $T' = T(c_i)$ be a subtree of error tree T and $\Delta > 0$ be a given error bound. The shift range of c_i (i.e., range(c_i)) is defined as follows.*

a. *if T' is $\underline{s}$-unbound, then range$(c_i) = [\underline{s_i}, \overline{s_i}]$ where $\underline{s_i} = [(x_L - x_R) - |l_L - l_R|]/2$ and $\overline{s_i} = [(x_L - x_R) + |l_L - l_R|]/2$ derived from Equation (7).*

b. *if $T' = T(c_0)$ is s-bound but not Δ-bound, range$(c_0) = (\underline{s_0}, \overline{s_0})$ where $\underline{s_0} = x_1 - (\Delta - l_1)$ and $\overline{s_0} = x_1 + (\Delta - l_1)$ derived from Equation (5).*

c. *if T' is s-bound, then range$(c_i) = [\underline{s_i}, \overline{s_i}]$ where $\underline{s_i} = \overline{s_i} = (x_L + x_R)/2$.*

As we will explain in the latter sections, the S-Shift algorithm will employ "modified" shift ranges in the process of generating a synopsis. Clearly, $s_i = (x_L - x_R)/2$, which will be used in the F-Shift algorithm constantly, is a solution of Equation (6).

EXAMPLE 3.10. *In Figure 1(iii), range$(c_1) = [5.5, 6]$ from Equation (7). That is, $T\langle\{s_1\}\rangle$ is s-bound for any $s_1 \in$ range(c_1). The data interval $[d_s, d_g]$ of $T(s_1)$ slides between $[-1.5, 13.5]$ and $[-2, 13]$ for $s_1 \in$ range(c_1) with the same data scope $l_2 = 7.5$. In Figure 1(iv), T is s-bound. range$(c_0) = (5.25, 6.25)$ by Equation (5). $T\langle\{s_0\}\rangle$ is Δ-bound for any $s_0 \in$ range(c_0). Data interval (d_s, d_g) of $T\langle\{s_0\}\rangle$ slides between $(-7, 8)$ and $(-8, 7)$. Specifically, $T\langle\{s_0\}\rangle$ for $s_0 = 5.5$ is the error tree of Figure 1(v).*

3.4 Idea on Shift Algorithms

The properties of shift transformations, data scopes and shift ranges will be used to shape the Shift algorithms. Normally, Shift algorithms work bottom-up from the data vector. It applies a shift transformation on the root node of a current $\underline{s}$-unbound subtree repeatedly. Each shift transformation converts the $\underline{s}$-unbound subtree into a s-bound subtree through properly chosen a shift value from the shift range. When the process comes to node c_0, the algorithms may shift node c_0 making the current $T(c_0)$ be Δ-bound. Eventually, the set of shift coefficients is the synopsis. For instance, let us consider the error tree T of Figure 1(ii). The F-Shift algorithm first works on the $\underline{s}$-unbound subtree $T(c_5)$ and derives $T\langle\{s_5\}\rangle$ depicted in Figure 1(iii). It then works on the next $\underline{s}$-unbound subtree, which is $T(c_1)$ in Figure 1(iii), and derives $T\langle\{s_1, s_5\}\rangle$ depicted in Figure 1(iv). Repeating these steps, the F-Shift algorithm derives $T\langle\{s_0, s_1, s_5\}\rangle$ ultimately.

The major difference between the F-Shift and the S-Shift is that they use different strategies on choosing shift values. Unlike the F-Shift where the shift values are set to be average values (i.e., $(x_L -$

$x_R)/2$) from bottom up, the S-Shift algorithm generate synopsis S_S from two phases: the bottom-up phase and the top-down phase. In the bottom-up phase, it computes *range*(c) from *range*(c_L) on the left subtree and *range*(c_R) on the right subtree in a "modified" way by generalized Equation (7) and Definition 3.9. Its objective is to fuse the maximum overlapped data range on each subtree. In the top-down phase, it instantiates the values of selected shift coefficients within the ranges to compute synopsis S_S from top down. Details will be given in the next two sections.

4. F-SHIFT ALGORITHM

As depicted in Algorithm 1, the F-Shift (Fixed-value Shift) algorithm employs the data scope and shift technique to obtain a synopsis in bottom-up from the data vector. It does not need to construct W_D or T previously. This is because that a shift transformation on $c \in T$ actually only works on data of $T_L(c)$ and $T_R(c)$ in an opposite way and can be described by the data "intervals" of $T_L(c)$ and $T_R(c)$.

In this section, we first describe the F-Shift algorithm. We then study the complexity of F-Shift and discuss the upper bound on the synopses generated from it.

Algorithm 1 : F-Shift(D,Δ)

Input:
D, the original data vector; Δ, the specified error bound.
Output:
A synopsis B. That is, the set of shift coefficients that satisfies Δ bound.
Description:
1: B=$\emptyset$; F=$\emptyset$ {Initialize the synopsis and set F}
2: define $f = (x, l, n)$ {x and l describe the current shift range; n is the number of data in the current subtree}
3: **for** $d_i \in D$ **do**
4: $f = (d_i, 0, 1)$; add f to F
5: **while** there exist $f_l = (x_l, l_l, n_l)$ and $f_r = (x_r, l_r, n_r)$ in F, such that $n_l = n_r$ **do**
6: set $d_g = \max\{x_l+l_l,\ x_r+l_r\}$ and $d_s = \min\{x_l-l_l,\ x_r-l_r\}$
7: **if** $(d_g - d_s \geq 2\Delta)$ **then**
8: set $b=(x_l-x_r)/2$, $l=max\{l_l,\ l_r\}$ and $x=(x_l-b)$
9: **else**
10: set $b = 0$, $l=(d_g-d_s)/2$ and $x=(d_g+d_s)/2$
11: **end if**
12: add $f = (x, l, 2n_r)$ to F
13: delete f_l and f_r from F
14: if $b \neq 0$, add b to B
15: **end while**
16: **end for**
Ensure: there is only one element, $f = (x, l, n)$, in F
17: if $|x| \geq |\Delta - l|$, add x to B
18: **return** B

4.1 Description of F-Shift

Roughly, the F-Shift algorithm constructs a synopsis from d_0 up to d_{N-1} gradually onwards (lines 3). This progressive feature is desirable in on-line processing of stream data.

Line $(7-11)$, is the key part of the algorithm. It uses a "fixed" value to convert an $\underline{s}$-unbound subtree into an s-bound one. It constructs $f = (x, l, 2n)$ from $f_L = (x_L, l_L, n)$ on the left subtree and $f_R = (x_R, l_R, n)$ on the right subtree. Suppose that d_{L_g} and d_{L_s} are the largest and smallest data values on the left subtree. Each data in the subtree is in the interval of (d_{L_s}, d_{L_g}). In $f_L = (x_L, l_L, n, b_L)$, $x_L=(d_{L_g}+d_{L_s})/2$ is the mean of d_{L_g} and d_{L_s}, $l_L=(d_{L_g}-d_{L_s})/2$ is the data scope, n is the number of data in the subtree.

If the current subtree is $\underline{s}$-unbound, a shift transformation is required (Line 8). Let the shift value $b=(x_L-x_R)/2$. Clearly, such b satisfies Equation (7). Line 17 decides s_0, the shift value of c_0. This step is based on Equation (5). If $|x_1| \geq \Delta - l$ then shift c_0 into x_1 that satisfies Equation (5).

Property 3.1 guarantees that Line $(5-15)$ works on an error tree at each iteration. Property 3.3 assures that the final B of the F-Shift algorithm is a Δ-bound synopsis. The value of each shift coefficient is assigned according to the designated shift range.

EXAMPLE 4.1. *Considering error tree T of Figure 1(ii), F-shift algorithm generates $s_5 = 8$, $s_1 = 5.75$ and $s_0 = 5.75$ incrementally and eventually obtains the synopsis $S_F = \{s_5, s_1, s_0\}$.*

4.2 Complexity and Upper Bound

In the F-Shift algorithm, the parameters on a node are derived from the parameters on the two children nodes with a constant number of operations. Since there are N number of nodes that need to be processed at most, the total time required to compute B (i.e., S_F) is $O(N)$.

As the algorithm works in bottom up from the data vector, the space complexity of F-Shift is $O(\log N)$. This is because that each s_i of S_F needs to be derived from the information on nodes $\{c_{2i}, c_{2^2i+2}, c_{2^3i+2^2+2}, ...\}$, one node at each level of $T(c_i)$. That is, $\log N$ in number.

To estimate $|S_F|$, let the number of $\overline{s}$-bound subtrees of T be t_s. Since the F-Shift algorithm needs to shift the root node of two adjacent $\overline{s}$-bound subtrees to make it s-bound, it needs to shift at most $t_s - 1$ nodes to make T become s-bound. Again the F-Shift algorithm may need to adjust c_0 node at the last step to make T Δ-bound. Therefore, $|S_F| \leq 1 + t_s - 1$. That is, $|S_F| \leq t_s$. Thus the following lemma is proven.

LEMMA 4.2. *Let the number of $\overline{s}$-bound subtrees of T is t_s. Then $|S_F| \leq t_s$.*

Moreover, the above bound on S_F is "tight": it can easily find a example such that $|S_F| = t_s$. However, this bound is not applicable to restricted-optimal synopsis M_{op}: It can be either $|M_{op}| \leq t_s$ or $|M_{op}| > t_s$.

EXAMPLE 4.3. *In Figure 1(ii), since $t_s = 4$ and $|M_{op}| = 5$, $|M_{op}| > t_s$ hold. In Figure 2(i), since $t_s = 4$ and $|M_{op}| = 2$, $|M_{op}| < t_s$ hold.*

Furthermore, we have $|S_F| \leq |S_{op}| \log N$. This is because, for each c_i that satisfies $|T(c_i) \sqcap S_F| = 1$, there exist s_j such that $s_j \in S_{op} \sqcap T(c_i)$ (Property 3.6(iv)). Let S_J be the set of all s_j. Then $S_J \subseteq S_{op}$. As the number of ancestors of s_j is bounded by $\log N$, $|S_F| \leq |S_J| \log N \leq |S_{op}| \log N$ holds. In summary, we have proven the following theorem for this section.

THEOREM 4.4. *Let D be a data vector with size N and $\Delta > 0$. Suppose S_F is the set of shift coefficients obtained from F-Shift algorithm on D and Δ. Then*

- *S_F is a synopsis;*
- *the time and space complexity of F-Shift algorithm are $O(N)$ and $O(\log N)$ respectively;*
- *the size of S_F is bounded by $\min\{t_s, |S_{op}| \log N\}$.*

EXAMPLE 4.5. *In Figure 2(i), it can be verified that $M_{op} = \{c_4, c_7\}$ and $S_F = \{s_0, s_2, s_3\}$ for $s_0 = 3.5$, $s_2 = 4$ and $s_3 = -4$. From Equation (7), range$(c_2) = [3, 5]$ and range$(c_3) = [-4.5, -3.5]$. It can also show that $S_{op} = \{s'_2, s'_3\}$ for $s'_2 = 5$ and $s'_3 = -4$ in Figure 2(ii).*

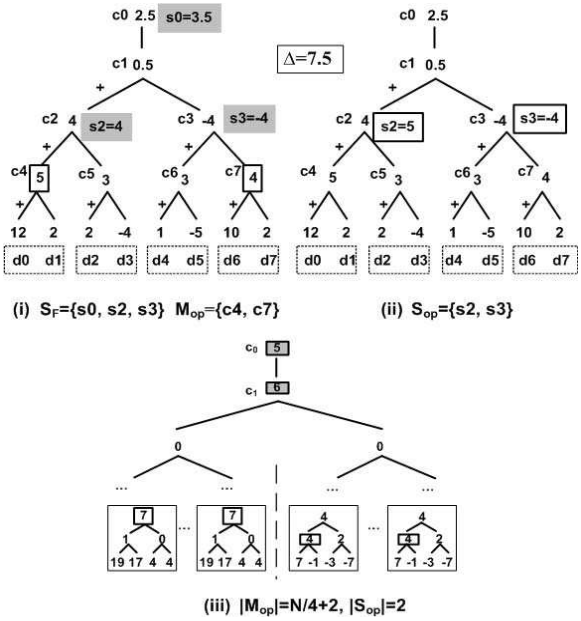


Figure 2: Example.

Besides $|S_F| > |S_{op}|$, the above example also indicates that choosing proper shift values can reduce the size of S_F. This topic will be discussed in Section 5. In contrast, as indicated in the following example, the size of R-optimal $|M_{op}|$ can be very large even if $|S_{op}|$ is quite small.

EXAMPLE 4.6. *As depicted in Figure 2(iii), D is the data vector that is formed by repeating $D_L = [19; 17; 4; 4]$ and $D_R = [7; -1; -3; -7]$ on its left and right subtree respectively. Let $\Delta = 8$. It can prove that $S_F = \{s_0 = 5.75, s_1 = 5.75\}$ for each $N = 2^n > 8$. It can also show that $|M_{op}| = N/4 + 2$. For instance, $M_{op} = \{c_0, c_1, c_4, c_5, c_{12}, c_{14}\}$ when $N = 16$.*

5. S-SHIFT ALGORITHM

The S-Shift (Sliding-value Shift) algorithm is aimed at constructing a smaller sized synopsis. As indicated in Formula (7) of Section 3.3, there exist many shift values other than $b = (x_L - x_R)/2$ that used for the F-Shift algorithm. As an extension of the F-Shift algorithm, S-Shift algorithm employs shift ranges and "late binding" on shift values to compute a synopsis to diminish the number of shift coefficients. In particular, the strategy behind S-Shift is that first find the set of coefficients that need to be shifted along with their fused shift ranges from bottom-up and then instantiate a proper shift value for each selected coefficient from its range to obtain a synopsis in top-down. In this way, some shift coefficients in S_F can be evaporated away through fusing the shift ranges on their subtrees. The detailed descriptions and the correctness proof for the algorithm, along with a running example, are given in Appendix.

5.1 The property of S-Shift

In the bottom-up phase of the S-Shift algorithm, each $f = (c, \_)$, derived from $f_L = (c_L, \_)$ and $f_R = (c_R, \_)$ with a bounded number of operations, requires $O(1)$ time. Since there are N number of nodes that need to be processed, the time required in this phase is $O(N)$. The space complexity of this phase is also $O(N)$ as each $f = (c, \_)$ is required in the top-down phase. In the top-down phase, each $f = (c, \_)$ is used either to instantiate a value for an element of B or to narrow down shift ranges. This phase requires $O(N)$ in both memory and time for processing N number of nodes. Therefore, the total time/memory requirement of the S-Shift algorithm is $O(N)$.

With regard to the size of S_S, the synopsis constructed from the S-Shift algorithm relates a varied form of U-optimal problem, which is called *U-scope-based-optimal* problem that stated as follows.

DEFINITION 5.1. *Given data vector $D = [d_0; d_1; ...; d_{N-1}]$ and error bound Δ, the U-scope-shift-optimal (U_s-optimal) problem is to find S_{opt}, a synopsis with the smallest set of unrestricted coefficients among all possible solutions, such that*

i *S_{opt} is a synopsis on Δ. That is, $|d_i - \hat{d}_i^{(S_{opt})}| < \Delta$ holds for $0 \le i \le N-1$;*

ii *$T_i\langle S_i\rangle$ is $\underline{s}$-unbound for each $s_i \in S_{opt}$ where $S_i = S_{opt} - \{s_i\}$ and $T_i = T(c_i)$ and $i > 0$.*

Roughly, the difference between U-optimal problem and U_s-optimal problem is that s_i can be chosen from a s-bound subtree in U-optimal problem but cannot be so in U_s-optimal problem.

As mentioned early, Phase A minimizes the scope of each subtree and uses Equation (7) rather than Equation (4) to compute ranges. One issue of concern is that this approach of minimization may result in the lose of generality (i.e., a smaller sized synopsis may exist if uses Equation (4)). The concern is clarified in Property 5.2.

PROPERTY 5.2. *Let $T_{i_L} = T_L(c_i)$ and $T_{i_R} = T_R(c_i)$ be the left and right substree of $T_i = T(c_i)$.*

1. *If $s_i \in S_{opt}$ and s_i does not lead to the minimal scope of $T_i\langle S_{opt}\rangle$ from $T_{i_L}\langle S_{opt}\rangle$ and $T_{i_R}\langle S_{opt}\rangle$. Then there exists s_i' that can minimize the scope of $T_i\langle S_{opt}\rangle$ such that $(S_{opt} - \{s_i\}) \cup \{s_i'\}$ is a synopsis.*
2. *Suppose coefficient c_i is not shifted (i.e., $s_i \notin S_{opt}$). Let $S = T(c_i) \sqcap S_{opt}$, $S_L = T_L(c_i) \sqcap S_{opt}$ and $S_R = T_R(c_i) \sqcap S_{opt}$. We use d_s and d_g (d'_{L_s} and d'_{L_g}, d'_{R_s} and d'_{R_g}, respectively) to denote the smallest and largest data of $T_i\langle S\rangle$ ($T_L\langle S'_L\rangle$, $T_R\langle S'_R\rangle$, respectively) where S'_L (or S'_R) is obtained from S_L (or S_R) by assigning different values to some of its elements. If $d_s \le \min\{d'_{L_s}, d'_{R_s}\}$ and $\max\{d'_{L_g}, d'_{R_g}\} \le d_g$, then $(S_{opt} - S_L) \cup S'_L$, $(S_{opt} - S_R) \cup S'_R$ and $(S_{opt} - S_L \cup S_R) \cup S'_L \cup S'_R$ are synopses.*

PROOF. We only prove (1) as (2) can be proven similarly.

Let d_g and d_s (d_{L_g} and d_{L_s}, or d_{R_g} and d_{R_s}) be the largest and smallest data values of $T_i\langle S_{opt}\rangle$ ($T_L\langle S_{opt}\rangle$ or $T_R\langle S_{opt}\rangle$) respectively where $T_i = T(c_i)$. Since s_i is not maximum overlapping the scopes of $T_L\langle S_{opt}\rangle$ and $T_R\langle S_{opt}\rangle$, $(d_{L_g} - s_i > d_{R_g} + s_i) \wedge (d_{L_s} - s_i > d_{R_s} + s_i)$ or $(d_{L_g} - s_i < d_{R_g} + s_i) \wedge (d_{L_s} - s_i < d_{R_s} + s_i)$ holds.

If $(d_{L_g} - s_i > d_{R_g} + s_i) \wedge (d_{L_s} - s_i > d_{R_s} + s_i)$, let $s_i' = \min\{d_{L_g} - d_{R_g}, d_{L_s} - d_{R_s}\}$. It can be proven that

$$\begin{cases} \max\{(d_{L_g} - s_i'), (d_{R_g} + s_i')\} \le \max\{(d_{L_g} - s_i), (d_{R_g} + s_i)\}, \\ \min\{(d_{L_s} - s_i), (d_{R_s} + s_i)\} \le \min\{(d_{L_s} - s_i'), (d_{R_s} + s_i')\}. \end{cases}$$

That is,

$$\begin{cases} \max\{(d_{L_g} - s_i'), (d_{R_g} + s_i')\} \le d_g, \\ d_s \le \min\{(d_{L_s} - s_i'), (d_{R_s} + s_i')\}. \end{cases}$$

Let $d'_g = \max\{(d_{L_g} - s_i'), (d_{R_g} + s_i')\}$ and $d'_s = \min\{(d_{L_s} - s_i'), (d_{R_s} + s_i')\}$. The above formulae imply that $(|d'_g - \gamma| <$

$\Delta) \wedge (|d'_s - \gamma| < \Delta)$ if $|d_g - \gamma| < \Delta$ and $|d_s - \gamma| < \Delta$. Therefore, $(S_{opt} - \{s_i\}) \cup \{s'_i\}$ is a synopsis by definition.

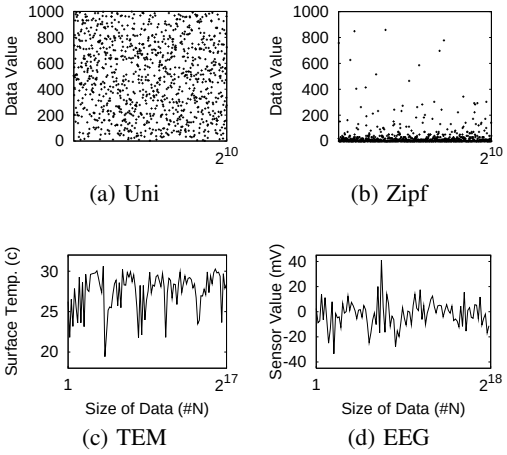
Similarly, it can be proven that the result for the situation of $(d_{L_g} - s_i < d_{R_g} + s_i) \wedge (d_{L_s} - s_i < d_{R_s} + s_i)$. $\square$

Intuitively, Property 5.2 means that S_{opt} can always be constructed by minimizing the scopes of subtrees. Since the S-Shift algorithm checks all possible shift values that lead to the minimized scopes of subtrees, the following theorem is proven.

THEOREM 5.3. *Let D be a data vector with size N and $\Delta > 0$. Suppose S_S is the set of shift coefficients obtained from S-Shift algorithm on D and Δ. Then*

- *S_S is an optimal synopsis for the U_s-optimal problem (i.e., $|S_S| = |S_{opt}|$ and $|S_S| \leq |S_F|$);*
- *the time and space complexity of S-Shift algorithm are $O(N)$;*
- *the size of S_S is bounded by $\min\{t_s, |S_{op}| \log N\}$.*

In the above theorem, the bound on $|S_S|$ is directly obtained from Theorem 4.4.



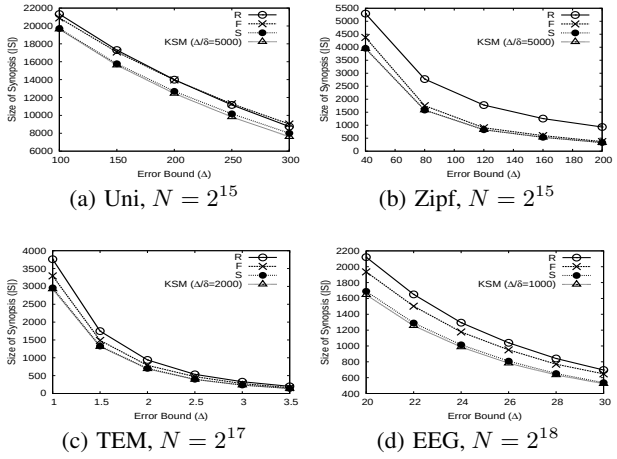
(a) Uni (b) Zipf (c) TEM (d) EEG

Figure 3: Distributions of various Data Sets

6. EXPERIMENTAL EVALUATION

In this section, we present some of our experimental results of the Shift algorithms to illustrate their compression quality and time efficiency. We compare the results to those achieved with the restricted error bound synopses construction algorithm in [12] and the KSM algorithm in [10]. We also test the scalability and workability on the Shift algorithms. All the algorithms are implemented in GNU C++ and all the experiments are performed on an Intel Xeon 3.0GHZ Linux box with 2 GB memory.

Experiment Bed[4]. The experiments are conducted on two synthetic data sets and two real life data sets. The two synthetic data sets, as depicted in Figure 3 (a) and (b), are *Uni* and *Zipf* respectively with their values distributed in $[0, 1000]$. Uni contains values following uniform distribution. Zipf contains values following zipf distribution with a zipf parameter of 2.0, i.e. a highly skewed distribution. The two real data sets used in this section are acquired from UCI KDD Archive [1]: *TEM* and *EEG* as depicted in Figure 3 (c) and (d) respectively. *TEM* is sea surface temperatures extracted from the El Nino Data set, taken from a series of buoys positioned throughout the equatorial Pacific. This data set is also used in [10] for its experiments. *EEG* includes values, in microvolt (mV), acquired from a large study to examine EEG correlates of genetic predisposition to alcoholism. We implement our two $(\log N)$-approximate algorithms, the F-Shift algorithm (F) and the S-Shift algorithm (S). As a reference, we also implement the optimal restricted error bound synopses construction algorithm [12], denoted by *R*. For the KSM algorithm, we use the source code that borrowed from the authors of [10].



(a) Uni, $N = 2^{15}$ (b) Zipf, $N = 2^{15}$ (c) TEM, $N = 2^{17}$ (d) EEG, $N = 2^{18}$

Figure 4: Sizes of Synopses on various Data Sets

6.1 Compression Quality

Since the size of synopses reflect the compression quality, we shall construct error bound synopses for each of the algorithms F, S, and R and compare these obtained synopses against the optimal one in this subsection. As stated in [10] that a smaller δ in KSM algorithm leads to a smaller sized synopsis under a sacrifice of construction time, we shall use the synopses generated from KSM algorithm under smaller δ as the substitutes of optimal synopses. With this intention in mind, we tune δ such that the average construction time on synopses by KSM is between 30 to 80 hours for each considered data set and assume these generated synopses are quite close to the optimal in the experiments.

Figure 4 presents the experiment results on the four data sets. The value of Δ/δ indicates how we set δ for KSM in each data set. In Figure 4, we choose the displayed range of error bounds (Δ) to be practical in real situations[5]. For example, on TEM data, the error bounds are set from 1 to 3.5 due to the small standard deviation (1.91). Larger error bounds can seriously blur the details of TEM and are quite undistinguishable.

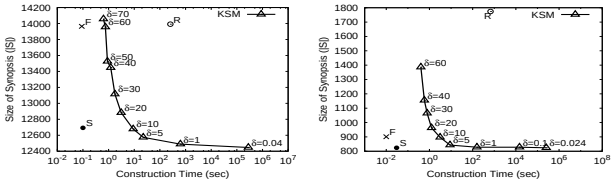
As indicated in Figure 4, F and S achieve better compression quality than that of R. S performs quite well since it generates similar sized synopses as that of KSM in many cases. S also performs better than F, which supports our previous theorems that S generates optimal shift synopses. We conclude that the two shift algorithms, especially S-shift, demonstrate superb compression quality.

[4]We also generate another moderate skew data set with values following normal distribution in our experiments. The test results on this data set lies between that of Uni and Zipf. Due to space limitation, we will not report the details in this version of this paper.

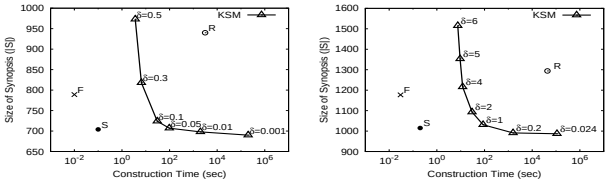
[5]Similar trends are discovered for other ranges

	Uni	Zipf	TEM	EEG
R/KSM	341	-	625	4367
F/KSM	0.12	3×10^{-3}	5×10^{-4}	1×10^{-3}
S/KSM	0.01	1×10^{-7}	4×10^{-4}	7×10^{-4}

Table 2: Synopses construction time comparison



(a) Uni, $N = 2^{15}$, $\Delta = 200$ (b) Zipf, $N = 2^{15}$, $\Delta = 120$



(c) TEM, $N = 2^{17}$, $\Delta = 2$ (d) EEG, $N = 2^{18}$, $\Delta = 24$

Figure 5: Time of synopses construction on various Data Sets

6.2 Time Efficiency

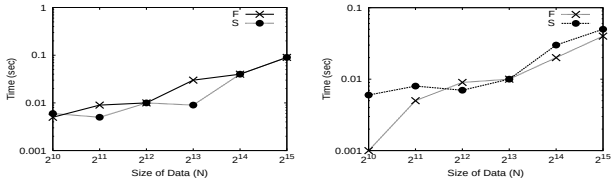
Unlike the running time of F, S and R that solely depends on N, the running time of KSM is also heavily related to $(\Delta/\delta)^2$: A big δ results in a quick construction with poor compression quality while a small δ results in a slow construction with a good compression quality. In this subsection, we evaluate the synopses construction time under a fixed error bound (Δ) and various δ values on KSM. From the following discussion, we conclude that the two shift algorithms, especially S-shift one, can be $1 \sim 7$ orders of magnitudes faster than KSM algorithm when constructing similar sized synopses.

Figure 5 shows the synopses construction time on the four data sets. KSM demonstrates a trade-off between construction time and synopses sizes. Table 2 lists the construction time ratios with KSM upon constructing similar sized synopses. It indicates that R is far more slower than KSM on constructing a similar quality synopsis. The shift algorithms, F and S both perform better than KSM under a smaller δ. That is, when constructing similar sized synopses, F improves the synopses construction time $1 \sim 4$ orders of magnitudes. S improves the synopses construction time $2 \sim 7$ orders of magnitudes. This can be explained from $O((\frac{\Delta}{\delta})^2 N)$ time complexity on smaller δ. Besides, termed as TM data set, the TEM data set is also used in Figure 8(a) of [10]. When constructing synopses for $\delta = 0.1$ on the TEM data set, Figure 5(c) shows that KSM requires no more time than that of indicated in Figure 8(a) of [10].

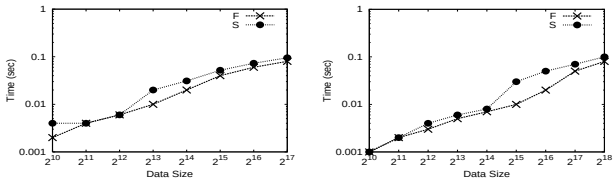
It is also interesting to note that the performances of F and S are especially well when constructing synopses on data sets with skewed distribution.

6.3 Scalability

To evaluate the time and compression quality performance trends of F and S on large data sets, the scalability test is twofold: (A) evaluating the synopses construction time on various data sizes and (B) assessing the compression quality (synopses size) on various data sizes. For the KSM in (B), we set small δ values as we did in section 6.1 to generate near optimal results. We omit the tests of R in this part.



(a) Uni, $\Delta = 200$,$\delta = 0.04$ (b) Zipf, $\Delta = 120$,$\delta = 0.024$

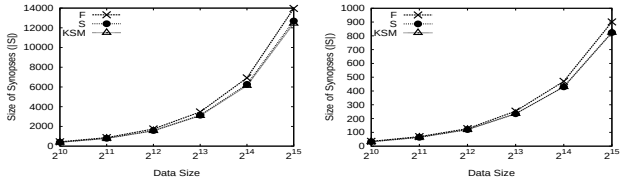


(c) TEM, $\Delta = 2$,$\delta = 0.001$ (d) EEG, $\Delta = 24$,$\delta = 0.024$

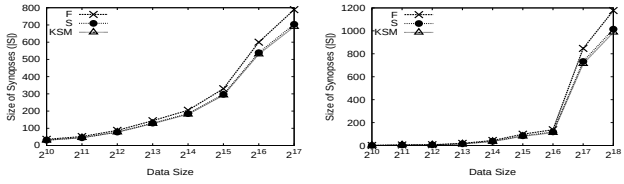
Figure 6: Scalability on synopses construction

Figure 6 presents the results for (A). It shows that the running time of F and S keeps a straight line and scales well with larger data sizes. In these tests, both shift algorithms can generate arbitrary error-bound synopses in less than 1 second, even for very large data sets with up to 2^{18} values.

Figure 7 presents the results for (B). It shows that the shift algorithms have good compression quality on various data sizes. The S performs especially well as we can hardly distinguish it from the near optimal results generated by KSM in all our experiment results.



(a) Uni, $\Delta = 200$,$\delta = 0.04$ (b) Zipf, $\Delta = 120$,$\delta = 0.024$

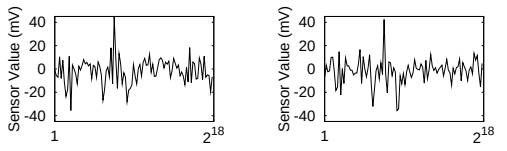


(c) TEM, $\Delta = 2$,$\delta = 0.001$ (d) EEG, $\Delta = 24$,$\delta = 0.024$

Figure 7: Scalability on compression quality

6.4 Workability

Our last test focuses on the workability of our two shift-based algorithms. As mentioned early, one application of our techniques is to compress the time series data acquired through clinical operations. The generated synopses will be used to reconstruct the



(a) $\Delta = 5, |S| = 28168$ (b) $\Delta = 20, |S| = 1690$

Figure 8: Approximation of the EEG Data set

original data values under some error bounds. We use the synopses generated by S to reconstruct the original data. Specifically, the synopses reconstruct the EEG data set. Figure 8 shows the reconstructed approximation data of the EEG data set where the two synopses have error bounds 5 and 20 respectively. Obviously, the two reconstructed data set are quite similar to the original data set, while the synopses size is only 10% to 0.6% of the original data size.

6.5 Summary

In general, all the experiments indicate that the Shift algorithms construct small size of synopses for both synthetic and real data sets. The S-Shift algorithm is especially efficient, in terms of synopses size. Moreover, both Shift algorithms construct synopses in less than 0.1 second, even when the size of original data reaches up to 2^{18}. This demonstrates that our algorithms are efficient and practically capable for large data sets.

7. CONCLUSION

In this paper, we have proposed two new algorithms for constructing the unrestricted error bound synopses. The algorithms are highly practical, cheap to run and generate smaller-sized synopses. Of the two shift algorithms, as we have previously mentioned, the F-Shift algorithm can compress stream data directly. In conclusion,

However, the paper leaves many attractive open problems to be answered in the future work. There exist gaps between F-Shift and S-Shift algorithms: If better compressing outcomes can be achieved through using other fixed values rather than the ones that are used in F-Shift? Besides, it is still not clear how to find U-optimal synopsis S_{op} more efficiently. Our future work will consider these problems.

It should be noted that Guha et al proposed new results on error-bound synopses [7] after this work was submitted for publication. The comparisons with our approaches will be presented in [17].

Acknowledgments: The authors would like to thank Panagiotis Karras, Dimitris Sacharidis and Nikos Mamoulis for their generous giving us their source code for the comparison tests in this paper.

8. REFERENCES

[1] KDD archive. http://kdd.ics.uci.edu.

[2] M. Garofalakis and P. B. Gibbons. Wavelet synopses with error guarantees. In *ACM SIGMOD*, pages 476–487, 2002.

[3] M. Garofalakis and A. Kumar. Deterministic wavelet thresholding for maximum-error metrics. In *ACM PODS*, pages 166–176, 2004.

[4] S. Guha. Space efficiency in synopsis construction algorithms. In *VLDB*, pages 409–420, 2005.

[5] S. Guha and B.Harb. Approximation algorithms for wavelet transform coding of data streams. In *SODA*, pages 698–707, 2006.

[6] S. Guha and B. Harb. Wavelet synopsis for data streams: minimizing non-euclidean error. In *ACM SIGKDD*, pages 88–97, 2005.

[7] S. Guha and B. Harb. Approximation algorithms for wavelet transform coding of data streams. *IEEE Transactions on Information Theory*, 54(2):811–830, Feb. 2008.

[8] S. Guha, K. Shim, and J. Woo. Rehist: Relative error histogram construction algorithms. In *VLDB*, pages 300–311, 2004.

[9] P. Karras and N. Mamoulis. One-pass wavelet synopses for maximum-error metrics. In *VLDB*, pages 421–432, 2005.

[10] P. Karras, D. Sacharidis, and N. Mamoulis. Exploiting duality in summarization with deterministic guarantees. In *ACM SIGKDD*, pages 380–389, 2007.

[11] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. In *ACM SIGMOD*, pages 448–459, 1998.

[12] S. Muthukrishnan. Subquadratic algorithms for workload-aware haar wavelet synopses. In *FSTTCS*, pages 285–296, 2005.

[13] C. Pang, Q. Zhang, D. Hansen, and A. Maeder. Constructing unrestricted wavelet synopses under maximum error bound. http://e-hrc.net/pubs/papers/pdf/shiftWaveletRelACM-1.pdf. Technical Report (08/209), ICT CSIRO.

[14] C. Pang, Q. Zhang, D. Hansen, and A. Maeder. Building data synopses within a known maximum error bound. In *APWeb/WAIM2007*, 2007.

[15] E. J. Stollnitz, T. D. Derose, and D. H. Salesin. *Wavelets for computer graphics: theory and applications*. 1996.

[16] J. S. Vitter and M. Wang. Approximate computation of multidimensional aggregates of sparse data using wavelets. In *ACM SIGMOD*, pages 193–204, 1999.

[17] Q. Zhang, C. Pang, and D. Hansen. On multidimensional wavelet synopses for maximum error bounds. CSIRO Technical Report.

APPENDIX

Appendix:

Algorithm 2 S-Shift(D,Δ)

Input:
D, the original data set; Δ, the specified error bound; T, the error tree.
Output:
B, the set of shift coefficients that satisfies Δ bound.
Description:
1: $B = \emptyset$ {Initialize the synopsis set}
2: **for** $d_i \in D$ **do**
3: $f_{N+i} = (c_{N+i}, d_i, d_i, 0, 0)$;
4: **end for**
5: {*Bottom-Up Phase A: Compute* B *and* $f_i = (c_i, \underline{x_i}, \overline{x_i}, l_i, g_i)$ *from* f_{2i} *and* f_{2i+1} *from bottom-up* (Refer to Section .1.1)}
6: {*Top-Down Phase B: Compute each value of* B, $f_{2i} = (c_{2i}, x_{2i}, x_{2i}, l_{2i}, g_{2i})$ *and* $f_{2i+1} = (c_{2i+1}, x_{2i+1}, x_{2i+1}, l_{2i+1}, g_{2i+1})$ *from* $f_i = (c_i, x_i, x_i, l_i, g_i)$ (Refer to Section .1.2)}
7: **return** B

.1 Description of S-Shift

Depicted in Algorithm 2, the S-Shift algorithm has two phases in function: Phase A (bottom-up phase) and Phase B (top-down phase). In Phase A, it marks the coefficients that need to be shifted and computes the reduced shift range from the ranges on the left and right subtrees. In Phase B, it locates a feasible shift value within the shift range for each marked coefficient generated in Phase A.

To obtain a synopsis[6], the S-Shift algorithm computes and stores $f = (c, \underline{x}, \overline{x}, l, g)$ in bottom-up for each coefficient c: l is the scope of the current $T(c)$; g is a flag indicator that specifies c need to shift if g is assigned to 1. The interval $[\underline{x}, \overline{x}]$ indicates that the data interval $[d_s, d_g]$ of $T(c)$ can slide from $[\underline{x}-l, \underline{x}+l]$ to $[\overline{x}-l, \overline{x}+l]$ and can be denoted as $[x-l, x+l]$ for $x \in [\underline{x}, \overline{x}]$ in general.

The detailed descriptions and proof for the two phases are given in the next two subsections.

.1.1 Phase A (line 5) description

The central part of this phase is to compute $f = (c, \underline{x}, \overline{x}, l, g)$ from left child $f_L = (c_L, \underline{x_L}, \overline{x_L}, l_L, g_L)$ and right child $f_R = (c_R, \underline{x_R}, \overline{x_R}, l_R, g_R)$ based on the following two facts:

(i) $l_L < \Delta$ and $l_R < \Delta$;

(ii) each data d_L of $T_L(c)$ is in interval $R(x_L) = [x_L - l_L, x_L + l_L]$ and each data d_R of $T_R(c)$ is in interval $R(x_R) = [x_R - l_R, x_R + l_R]$ for $x_L \in [\underline{x_L}, \overline{x_L}]$ and $x_R \in [\underline{x_R}, \overline{x_R}]$.

To obtain $f = (c, \_)$, the procedure first checks whether coefficient c need be shifted then computes the parameters accordingly. Clearly, coefficient c do not need to shift if and only if $\exists x_L, x_R$, where $x_L \in [\underline{x_L}, \overline{x_L}]$ and $x_R \in [\underline{x_R}, \overline{x_R}]$, such that[7]

$$\max_{\substack{d' \in R(x_L) \\ d'' \in R(x_R)}} |d' - d''| < 2\Delta. \tag{8}$$

Roughly, the idea of this phase is as follows:

1. If Formula (8) holds, obtain x such that $R(x)$ is enclosed in every $R(x_L) \cup R(x_R)$ that will be illustrated in Step (A2) and Step (A3) in the following.
2. Otherwise, shift coefficient c and obtain x such that $R(x_L - x)$ and $R(x_R + x)$ are enclosed that will be illustrated in Step (A1) in the following.

More specifically, the computation of $f = (c, \underline{x}, \overline{x}, l, g)$ from $f_L = (c_L, \underline{x_L}, \overline{x_L}, l_L, g_L)$ and $f_R = (c_R, \underline{x_R}, \overline{x_R}, l_R, g_R)$ works as follows:

A1. Compute parameters when c needs to be shifted. Coefficient c need to be shifted if

$$min|x_L - x_R| + (l_L + l_R) \geq 2\Delta. \tag{9}$$

This means that current $T(c)$ is $\underline{s}$-unbound no matter what values of $x_L \in [\underline{x_L}, \overline{x_L}]$ and $x_R \in [\underline{x_R}, \overline{x_R}]$ are assigned. In this situation, coefficient c need to be shifted into s. In order to minimize the scope of $T\langle s \rangle$, s should satisfy Formula (7). Furthermore, s need to accumulate the shift ranges from $T_L(c)$ and $T_R(c)$. Therefore, it need to replace s_i of Formula (7) with $x = x_L - s_i$ (or $x = x_R + s_i$) to acquire the shift range of x:

$$(x_L + x_R) - |l_L - l_R| \leq 2x \leq (x_L + x_R) + |l_L - l_R|.$$

By replacing $(x_L + x_R)$ with $(\underline{x_L} + \underline{x_R})$ on the left and with $(\overline{x_L} + \overline{x_R})$ on the right in the above formula,

$$(\underline{x_L} + \underline{x_R}) - |l_L - l_R| \leq 2x \leq (\overline{x_L} + \overline{x_R}) + |l_L - l_R|.$$

Based on this formula, set

$$\begin{cases} \underline{x} = [(\underline{x_L} + \underline{x_R}) - |l_L - l_R|]/2, \\ \overline{x} = [(\overline{x_L} + \overline{x_R}) + |l_L - l_R|]/2, \\ l = \max\{l_L,\ l_R\} \text{ and g=1.} \end{cases}$$

A2. Compute parameters when c does not need to shift and

$$|l_L - l_R| < min|x_L - x_R| < 2\Delta - (l_L + l_R)$$

holds. In this situation, although $T(c)$ is s-bound, there exist a unique maximum overlapped data range (i.e., $\underline{x} = \overline{x}$) and the minimized scope of $T(c)$ (through adjusting x_L and x_R) is large than $\max\{l_L, l_R\}$. Set

$$l = (l_L + l_R + min|x_L - x_R|)/2 \text{ and } g = 2.$$

Since the upper data interval needs to move downwards and the lower interval needs to move upwards, two cases exist:

- If $\underline{x_L} > \underline{x_R}$, set $\underline{x}$ and $\overline{x}$ to be $(l_L + \underline{x_L} + \overline{x_R} - l_R)/2$ as $x + l = l_L + \underline{x_L}$ and $x - l = \overline{x_R} - l_R$.
- If $\underline{x_L} \leq \underline{x_R}$, set $\underline{x}$ and $\overline{x}$ to be $(l_R + \underline{x_R} + \overline{x_L} - l_L)/2$ as $x + l = \underline{x_R} + l_R$ and $x - l = \overline{x_L} - l_L$.

A3. Compute parameters when c does not need to shift and

$$min|x_L - x_R| \leq |l_L - l_R|$$

holds. Under this situation, the maximum overlapped range can be located in many places (i.e., $\underline{x} < \overline{x}$). Set

$$l = \max\{l_L,\ l_R\} \text{ and } g = 3.$$

If $l_L > l_R$, move the shorter scope (l_R) to be included by the longer scope (l_L). Using x to replace x_L in the above formula,

$$\begin{cases} -|l_L - l_R| \leq x - x_R \leq |l_L - l_R|, \\ \underline{x_L} \leq x \leq \overline{x_L}. \end{cases}$$

This implies,

$$\max\{\underline{x_L},\ \underline{x_R} - |l_L - l_R|\} \leq x \leq \min\{\overline{x_L},\ \overline{x_R} + |l_L - l_R|\}.$$

Similarly, if $l_L \leq l_R$,

$$\max\{\underline{x_R},\ \underline{x_L} - |l_L - l_R|\} \leq x \leq \min\{\overline{x_R},\ \overline{x_L} + |l_L - l_R|\}.$$

Therefore,

1. if $l_L > l_R$, set
$$\begin{cases} \underline{x} = \max\{\underline{x_L}, \underline{x_R} - |l_L - l_R|\}, \\ \overline{x} = \min\{\overline{x_L},\ \overline{x_R} + |l_L - l_R|\}. \end{cases}$$
2. If $l_L \leq l_R$, set
$$\begin{cases} \underline{x} = \max\{\underline{x_R},\ \underline{x_L} - |l_L - l_R|\}, \\ \overline{x} = \min\{\overline{x_R},\ \overline{x_L} + |l_L - l_R|\}. \end{cases}$$

.1.2 Phase B (line 6) description

The objective of this phase is to assign a proper shift value[8] for each $\{c_i | f_i = (c_i, \underline{x_i}, \overline{x_i}, l_i, g_i) \wedge (g_i = 1)\}$ and narrow down the range $[\underline{x_i}, \overline{x_i}]$ into an instance value x_i of $\underline{x_i} \leq x_i \leq \overline{x_i}$ from top-down starting at c_0.

First, it computes s_0, the shift coefficient at c_0 and an instance value from $[\underline{x_1}, \overline{x_1}]$ to satisfy the selected s_0. By replacing x_1 with $\underline{x_1}$ (and $\overline{x_1}$) in Formula (5), it obtains

$$\underline{x_1} - (\Delta - l_1) < s_0 < \overline{x_1} + (\Delta - l_1).$$

If s_0's range includes 0, then c_0 does not need to shift and let $s_0 = 0$. Otherwise, set $s_0 = (\underline{x_1} + \overline{x_1})/2$. It can prove that $|x_1 - s_0| < |l_1 - \Delta|$. Add s_0 into B and set $sum(x) = s_0$.

[6]For easy descriptions, in S-Shift algorithm, we add c in $f(c, \_)$ and use $[\underline{x}, \overline{x}]$ rather than $[\underline{s}, \overline{s}]$ as in Definition 3.9.

[7]Or, equivalently, $|x_L - x_R| + (l_L + l_R) < 2\Delta$ as expressed similar to Formula (9).

[8]There may exist many feasible shift values for c_i. Without loss the generality, we assign a "middle" value in the range in our approach.

With the instantiated s_0, we need to instantiate data range $[\underline{x_1}, \overline{x_1}]$ into x to satisfy the selected s_0. Such x should satisfy,

$$\begin{cases} -(\Delta - l_1) < x < (\Delta - l_1), \\ \underline{x_1} - s_0 \leqslant x \leqslant \overline{x_1} - s_0. \end{cases}$$

In the above two formulas, the first one comes from Formula (5) by replacing $x_1 - s_0$ with x while the second means that $x_1 = x + s_0$. Set

$$\underline{x} = \overline{x} = (\max\{-|l_1 - \Delta|\,,\ \underline{x_1} - s_0\} + \min\{|l_1 - \Delta|\,,\ \overline{x_1} - s_0\})/2$$

Next, update $f_L = (c_L, \_)$, $f_R = (c_R, \_)$ into[9],

$$\begin{cases} f_L = (c_L, \underline{x_L} - sum(x), \overline{x_L} - sum(x), l_L, g_L\_), \\ f_R = (c_R, \underline{x_R} - sum(x), \overline{x_R} - sum(x), l_R, g_R\_). \end{cases}$$

Iteration Step: With the obtained $f = (c, \underline{x}, \overline{x}, l, g)$ and $sum(x)$, x_L in $f_L = (c_L, \_)$ and x_R in $f_R = (c_R, \_)$ will be instantiated from the following cases.

B1: When $f = (c, \underline{x}, \overline{x}, l, g) \wedge (g = 1)$ holds, Since $|x'_L - x'_R| < |l_L - l_R|$, $x'_L \in [\underline{x_L} - s_c, \overline{x_L} - s_c]$ and $x'_R \in [\underline{x_R} + s_c, \overline{x_R} + s_c]$,

$$\begin{cases} -|l_L - l_R| < x'_L - x'_R < |l_L - l_R|, \\ \underline{x_L} \leq x'_L + s_c \leq \overline{x_L}, \\ \underline{x_R} \leq x'_R - s_c \leq \overline{x_R}. \end{cases} \tag{10}$$

If $l_L \geq l_R$, set $x'_L = x$. First, derive the ranges of $(\underline{s_c},\ \overline{s_c})$ by Equation (10). Then set $s_c = (\underline{s_c} + \overline{s_c})/2$. Again from Equation (10) and s_c, derives $(\underline{x'_R},\ \overline{x'_R})$ and set $x'_R = (\underline{x'_R} + \overline{x'_R})/2$.

If $l_L < l_R$, set $x'_R = x$. Similar to the above, find x'_L and s_c.

B2: $f = (c, \underline{x}, \overline{x}, l, g) \wedge (g = 2)$ holds. Set $s_c = 0$, $x'_L = \underline{x_L}$ and $x'_R = \overline{x_R}$.

B3: $f = (c, \underline{x}, \overline{x}, l, g) \wedge (g = 3)$ holds. Set $v_c = 0$. Since $|x'_L - x'_R| < |l_L - l_R|$, $x_L = x'_L$ and $x_R = x'_R$,

$$\begin{cases} -|l_L - l_R| < x'_L - x'_R < |l_L - l_R|, \\ \underline{x_L} \leq x'_L \leq \overline{x_L}, \\ \underline{x_R} \leq x'_R \leq \overline{x_R}. \end{cases} \tag{11}$$

Similar to (**B1**), except that $s_c = 0$ in this case.

Next, process the children nodes after doing the following:

$$\begin{cases} \text{Add } s_c \text{ into } B \text{ if } s_c \neq 0, \\ \text{Set } sum(x_L) = sum(x) + s_c \text{ and } sum(x_R) = sum(x) - s_c, \\ \text{Set } f_L = (c_L, x'_L, x'_L, l_L, g_L) \text{ and } f_R = (c_R, x'_R, x'_R, l_R, g_R). \end{cases}$$

A Running Example: The idea of S-Shift algorithm is illustrated in the following on the example of Figure 2(i).

In Phase A: For each s-bound subtree $T(c_i)$, set $f_i = (c_i, (d_s + d_g)/2, (d_s + d_g)/2, (d_g - d_s)/2, 0)$ where d_g and d_s are the max and min data in $T(c_i)$ respectively. Therefore,

$$f_4 = (c_4, 7, 7, 5, 0), \qquad f_5 = (c_5, -1, -1, 3, 0),$$
$$f_6 = (c_6, -2, -2, 3, 0), \quad f_7 = (c_7, 6, 6, 4, 0).$$

Next, the algorithm generates f_i from f_{2i} and f_{2i+1}. Using f_4 and f_5 to generate $f_2 = (c_2, \underline{x_2}, \overline{x_2}, l_2, g_2)$, a shift coefficient s_2 is required as $T(c_2)$ is $\underline{s}$-unbound. g_2 is set to 1 indicating that c_2 needs to shift. As indicated in Example 4.5, $s_2 \in$ *range*(c_2) where *range*$(c_2) = [3, 5]$. For $s_2 \in$ *range*(c_2), the data interval $[d_s, d_g]$ of $T(s_2)$ slides from $[-3, 7]$ to $[-1, 9]$ with data scope $l_2 = (d_g - d_s)/2 = 5$. $[\underline{x_2}, \overline{x_2}]$ is used to describe the data interval of $[-1, 9] \vee [-3, 7] = [-3, 9]$ and is set to $[-3 + l_2, 9 - l_2] = [2, 4]$. As a result, $f_2 = (c_2, 2, 4, 5, 1)$. Similarly, $f_3 = (c_3, 1.5, 2.5, 4, 1)$ can be derived.

Now, it comes to compute $f_1 = (c_1, \underline{x_1}, \overline{x_1}, l_1, g_1)$ from f_2 and f_3. Coefficient c_1 will not need to be shifted if the data scope $l_1 < \Delta$. Furthermore, l_1 can described by $[x_2 - l_2, x_2 + l_2] \vee [x_3 - l_3, x_3 + l_3]$ where $x_2 \in [2, 4]$ and $x_3 \in [1.5, 2.5]$. To minimize $[x_2 - l_2, x_2 + l_2] \vee [x_3 - l_3, x_3 + l_3]$, l_1 is set to $\max\{l_2, l_3\} = 5$ as the data scope of $T(c_1)$ can not be shorter than $\max\{l_2, l_3\}$ and $x_1 \in [\underline{x_1}, \overline{x_1}]$ satisfies,

$$\begin{cases} -|l_2 - l_3| \leq x_1 - x_3 \leq |l_2 - l_3|, \\ \underline{x_2} \leq x_1 \leq \overline{x_2}. \end{cases}$$

Intuitively, the first equation in the above means $[x_3 - l_3, x_3 + l_3] \subseteq [x_1 - l_1, x_1 + l_1]$. The following is then derived:

$$\begin{cases} \underline{x_1} = \max\{\underline{x_2}, \underline{x_3} - |l_2 - l_3|\} = 2, \\ \overline{x_1} = \min\{\overline{x_2},\ \overline{x_3} + |l_2 - l_3|\} = 3.5. \end{cases}$$

Up to now, this phase has selected the set of shift coefficients $\{s_2, s_3\}$. Their specific values will be determined in Phase B.

In Phase B, First, it decides s_0, the shift coefficient at c_0 and an instance value from $[\underline{x_1}, \overline{x_1}]$ to satisfy the selected s_0.

By replacing x_1 with $\underline{x_1}$ (and $\overline{x_1}$) in Equation (5), it obtains

$$2 - (7.5 - 1) < s_0 < 3.5 + (7.5 - 1),$$

which is $s_0 \in (-4.5, 10)$. Since $0 \in (-4.5, 10)$, it means that c_0 does not need to shift. Next, it need to select $x \in [\underline{x_1} - s_0, \overline{x_1} - s_0]$ from

$$-|\Delta - l_1| < x < |\Delta - l_1|$$

to support the selected s_0 (Equation (5)). Clearly, an instance of the above equation is $\underline{x} = \overline{x}$, which is

$$(\max\{-|\Delta - l_1|,\ \underline{x_1} - s_0\} + \min\{|\Delta - l_1|,\ \overline{x_1} - s_0\})/2 = 2.25.$$

Thus, $f_1 = (c_1, 2.25, 2.25, 5, \_)$.

Similarly, the following equations is used to derive the instances of x_2 in f_2 and x_3 in f_3.

$$\begin{cases} -|l_2 - l_3| < x_2 - x_3 < |l_2 - l_3|, \\ \underline{x_2} \leq x_2 \leq \overline{x_2}, \\ \underline{x_3} \leq x_3 \leq \overline{x_3}. \end{cases} \Rightarrow \begin{cases} -1 < x_2 - x_3 < 1, \\ 2 \leq x_2 \leq 4, \\ 1.5 \leq x_3 \leq 2.5. \end{cases}$$

Set $x_2 = x_1 = 2.25$ as $l_2 > l_3$ and find one feasible solution is $x_3 = (1.5 + 2.5)/2 = 2$. Thus, $f_2 = (c_2, 2.25, 2.25, 5, 1)$ and $f_3 = (c_3, 2, 2, 4, 1)$.

To find the value of s_2 from $f_2 = (c_2, 2.25, 2.25, 5, 1)$, $f_4 = (c_4, 7, 7, 5, 0)$ and $f_5 = (c_5, -1, -1, 3, 0)$, we have the following equations:

$$\begin{cases} -|l_4 - l_5| < x_4 - x_5 < |l_4 - l_5|, \\ \underline{x_4} \leq x_4 + s_2 \leq \overline{x_4}, \\ \underline{x_5} \leq x_5 - s_2 \leq \overline{x_5}. \end{cases} \Rightarrow \begin{cases} -2 < x_4 - x_5 < 2, \\ x_4 + s_2 = 7, \\ x_5 - s_2 = -1. \end{cases}$$

Set $x_4 = x_2 = 2.25$ as $l_4 > l_5$ and find one feasible solution for s_2 is 4.75. Similarly, it can derive $s_3 = 4$.

[9] In fact, this step update f_2 and f_3.