Optimal Construction of Multi-Dimensional Indexes in Time-Series Databases: A Physical Database Design Approach

Sang-Wook Kim
Department of Computer, Information, and Communications Engineering
Kangwon National University, Korea

Jin-Ho Kim
Department of Computer Science
Kangwon National University, Korea

Byung-Ill Han
Department of Computer Science
Kangwon National University, Korea

Sang-Hyun Park
Department of Computer Science and Engineering, Pohang University of Science and Technology (POSTECH), Korea

Abstract

Similarity search in time-series databases is an operation that finds such data sequences whose changing patterns are similar to that of a query sequence. Typically, it hires the multi-dimensional index for its efficient processing. In order to alleviate the *dimensionality curse*, a problem in high-dimensional cases, the previous methods for similarity search apply the Discrete Fourier Transform(DFT) to data sequences, and take only the first two or three DFT coefficients for selecting organizing attributes of the multi-dimensional index. Other than this ad-hoc approach, there have been no research efforts on devising a systematic guideline for choosing the best organizing attributes among all the DFT coefficients. This paper first points out the problems occurred in the previous methods, and proposes a novel solution to construct the optimal multi-dimensional index. The proposed method analyzes the characteristics of a target database, and then identifies the organizing attributes having the best discrimination power. Finally, it determines the optimal number of organizing attributes by using a cost model for similarity search. We show the effectiveness of the proposed method through a series of experiments.

1 Introduction

A time-series database is a set of data sequences, each of which is an ordered list of elements. *Similarity search* is an operation that searches for the sequences or subsequences whose changing patterns are similar to that of a given query sequence [1, 2, 6, 7, 11]. It is of growing importance in many new applications such as data mining and data warehousing [11].

Most approaches for similarity search regard a sequence with n elements as a point in n-dimensional space, and define the similarity of two sequences by using the *Euclidean distance* between their corresponding points [1, 6, 7, 11]. They also use the *multi-dimensional index* such as the R-tree, R^*-tree, and R^+-tree for efficient handling of multi-dimensional points. To avoid the *dimensionality curse* [1] in a multi-dimensional index, they usually apply the Discrete Fourier Transform (DFT) to the data sequences, and select the first two or three DFT coefficients as organizing attributes of the multi-dimensional index [1, 6].

However, there have been no research efforts to devise a systematic guideline for selecting the best organizing attributes from a set of DFT coefficients. This paper mainly focuses on this issue. First, we point out the performance problem of the previous methods, and then propose a novel method for constructing the optimal multi-dimensional index as a solution. The proposed method analyzes the characteristics of a target time-series database, and then selects the organizing attributes with the high discrimination power. It also determines the optimal number of organizing attributes using the cost model of similarity search. We compare the proposed method with the previous ones to verify its effectiveness.

2 Motivation

2.1 Similarity Search

The similarity measure of any two sequences, $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_n)$ is the *Euclidean distance* $D(X, Y)$ [1, 6, 7, 11] defined as follows:

$$D(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Similarity search is defined as the operation that finds such data sequences Y whose Euclidean distances to the query sequence X are within a given distance tolerance.

2.2 Previous Methods

Typically, previous methods [1, 6, 2] for similarity search regard a sequence with n elements as a point in n-dimensional space, and utilize the multi-dimensional index such as the R-tree, R^*-tree, and R^+-tree for efficient searching. The storage and computation costs of a multi-dimensional index grow exponentially with n [4]; thus the multi-dimensional index suffers from serious performance degradation when n is large, which is general in a time-series database.

To overcome such a problem, the previous work usually depends on the dimensionality reduction techniques using the Discrete Fourier Transform (DFT) [2, 6]. The DFT converts the sequence $X = (x_1, x_2, ..., x_n)$ of *real-valued* elements in the time domain into the sequence $X_F = (x_{F_1}, x_{F_2}, ..., x_{F_n})$ of *complex-valued* elements in the frequency domain. Each complex-valued element x_{F_k} $(k = 1, 2, ...n)$ in the frequency domain is defined as follows:

$$x_{F_k} = \frac{1}{\sqrt{n}} \sum_{t=1}^{n} x_t \ exp(\frac{-j2\pi kt}{n})$$

where $j = \sqrt{-1}$. The Parseval's theorem proves that the energy of a sequence in the time domain is always identical to that of the same sequence in the frequency domain. Therefore, the DFT preserves the features of a sequence after the conversion. In addition, the first few DFT coefficients contain a majority of the energy of the sequence in the frequency domain.

Using these two properties, the previous methods [1, 6] take the first k $(k \ll n)$ DFT coefficients from the sequences after conversion and construct a multi-dimensional index with $2 * k$ coefficients as organizing attributes. Since k is much smaller than n, the storage and computation costs of a multi-dimensional index reduce significantly. However, this approach introduces the *false match* [1] due to the the energy loss caused by the DFT coefficients excluded for organizing attributes. Thus, we have to detect and discard such false matches in the final validation step.

2.3 Problems in Previous Methods

The cost of similarity search consists of two parts: the index access cost for index searching and the object access cost for resolving false matches. In case of large k, similarity search requires a small object access cost thanks to little energy loss, but a large index access cost due to the problem of the dimensionality curse. When k is small, on the contrary, the index access cost becomes smaller while the object access cost gets higher. Therefore, it is essential to devise a method that constructs an optimal multi-dimensional index by considering both object access and index access costs. The previous methods have the following problems in the sense of optimality in constructing their index structures.

1. They simply use the energy level of the DFT coefficients in choosing organizing attributes. We know that the organizing attributes should have discrimination power higher than others. High energy level, however, does not always imply high discrimination power.

2. They just recommend the first *two or three* DFT coefficients but do not provide a systematic guideline for determining the optimal number of organizing attributes.

3. They take both of the real and imaginary parts of a DFT coefficient together as organizing attributes. However, the real and the imaginary parts have different discrimination power even though they are from the same DFT coefficient.

3 The Proposed Method

3.1 Basic Idea

We propose to use the following strategies to solve the problems aforementioned.

1. The real and the imaginary parts are treated separately even though they are from the same DFT coefficient. For this separation, the new terminology, the *element coefficient* is defined as the real or imaginary part of a DFT coefficient. Thus, the task of choosing the organizing attributes is performed in the unit of element coefficients. By this separation, we can prevent any element coefficients of small discrimination power from participating in organizing attributes.

2. The *standard deviation* of the element coefficient is used as criteria in choosing organizing attributes. A large standard deviation of an element coefficient implies that a large number of sequences in a database have different values for that element coefficient. Thus, the element coefficients with large standard deviations will be selected as organizing attributes.

3. We analyze the characteristics of a target database in advance using a cost model. Through this analysis, we estimate the index and the object access costs, which are used subsequently to determine the optimal number of organizing attributes.

3.2 Cost Model

Because the cost of disk accesses typically dominates in a time-series database environment, we develop a cost

model that mainly considers the number of disk accesses. Assuming that the R*-tree is employed as an index structure, our cost model combines and extends the previous ones [5, 3] that use the *fractal dimension* D_0 and the *correlation fractal dimension* D_2.

Our cost model is based on the following equation. It becomes the fractal dimension when $q = 0$ and the correlation fractal dimension when $q = 2$. p_i is the number of points inside the i^{th} cell when we divide the multi-dimensional space into the hyper cubic grid cells of side r. These values are obtained in reality by the algorithms presented in [3].

$$D_q = \frac{1}{q-1}\frac{\delta \log \sum_i (p_i)^q}{\delta \log r} = constant$$

3.2.1 Index Access Cost

The index access cost is estimated by the following expression proposed in [5].

$$IndexAccessCost = \sum_{j=0}^{h-1} \frac{N}{{C_{eff}}^{h-j}} \prod_{i=0}^{k} (\sigma_j + \varepsilon)$$

Here, N is the total number of data sequences, and k is the number of organizing attributes participating in the multi-dimensional index. C_{eff}, the average number of entries per node, represents the effective capacity of a node of the R*-tree. h is the height of the R*-tree (the root is assumed at level $j = 0$ and the leaves at level $j = h-1$). ε is the distance tolerance given with a query sequence, and σ_j is the side of the square MBR of a node at level j. Based on the fractal dimension D_0, σ_j is computed by the expression:

$$\sigma_j = \left(\frac{{C_{eff}}^{h-j}}{N}\right)^{\frac{1}{D_0}}$$

3.2.2 Object Access Cost

The object access cost is estimated by the expression, a slightly modified version of the one proposed in [3].

$$ObjectAccessCost = \left(\frac{Vol(\varepsilon+\alpha,\odot)}{Vol(\varepsilon+\alpha,\square)}\right)^{\frac{D_2}{n}} * (N-1) * 2^{D_2} * (\varepsilon+\alpha)^{D_2}$$

where, D_2 is the correlation fractal dimension, n is the number of elements in a sequence, and ε is the distance tolerance. α is the sum of standard deviations of element coefficients not chosen as organizing attributes. $Vol(\varepsilon + \alpha, \square)$ is the volume of the hyper cubic cell of side $\varepsilon + \alpha$, and $Vol(\varepsilon + \alpha, \odot)$ the volume of the hyper sphere with radius $\varepsilon + \alpha$. N is the number of data sequences.

3.3 Algorithm

The detailed algorithm for constructing optimal multi-dimensional indexes is shown in Algorithm 1.

4 Performance Evaluation

4.1 Experimental Environment

The data and query sequences were generated using the expression used in [1]. First, using the expression $x_i = x_{i-1} + rand()$, we generated 10,000 data sequences $X = (x_i)$. Here, $rand()$ is the random function that picks the data value uniformly from the range of $[-500, 500]$. Every sequence has 8 elements ($n = 8$). To generate 1,000 query sequences $Q = (q_i)$, we used the expression $q_i = x_i + 0.05 * rand()$. Using $\sqrt{1000 \times n}$ as the base value ε_1 of a distance tolerance, we set ε_i as $\varepsilon_1 \times i$. The page size of the R*-tree was 512 bytes. We stored each sequence in a separate page to nullify the clustering effect.

4.2 Experimental Result

Figure 1 shows the experimental result. The X-axis is the distance tolerance submitted with a query sequence and the Y-axis is the average number of disk accesses for processing 1,000 query sequences. For a multi-dimensional index, the previous methods took the first 2 and 3 DFT coefficients, respectively while our method used the element coefficients chosen by the Algorithm 1. The figure shows that (1) the performance improvement of our method is significant (from 2 times to 100 times), and (2) the performance benefit gets larger when the distance tolerance becomes small. Therefore, the experimental results verified that the proposed method definitely has performance much better than the previous ones.

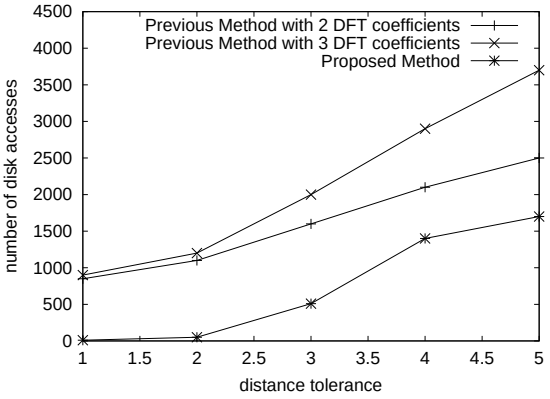

Figure 1: Performance comparisons of the proposed and previous methods in similarity search.

5 Conclusions

This paper proposed a unique method that enables to construct an optimal multi-dimension index for similarity search. First, it regards the real part and the imaginary

Input : Set of N data sequences with n elements, set of q sequences and a distance tolerance ε frequently used in queries

Output: Set of element coefficients selected as organizing attributes for a multi-dimensional index

1 Compute the fractal dimension D_0 and the correlation fractal dimension D_2 by analyzing all the data sequences stored in the database;
2 Access each data sequence from the database and perform the DFT;
3 Sort the element coefficients in the ascending order of their standard deviation, and insert the identifiers of the element coefficients into the array $EC(i)$ $(1 \leq i \leq 2n)$ in the sorted order;
4 **for** *each element coefficient* $EC(i)$ **do**
5 Suppose that an index is built by using i organizing attributes $EC(1), EC(2), ..., EC(i)$;
6 Given each query sequence $Q(j)$ $(1 \leq j \leq q)$ and ε, compute the index access cost $IndexAccessCost(i, j)$ and the object access cost $ObjectAccessCost(i, j)$ when utilizing the multi-dimensional index supposed in the line 5;
7 Using $IndexAccessCost(i, j)$ and $ObjectAccessCost(i, j)$, compute the total cost $TotalCost(i)$ for processing all the q queries;
8 Find $TotalCost(w)$, the smallest one among $TotalCost(i)$ $(1 \leq i \leq 2n)$;
9 Return element coefficients $EC(1), EC(2), ..., EC(w)$ as the organizing attributes of the optimal multi-dimensional index;

Algorithm 1: Constructing of an optimal multi-dimensional index.

part of a complex DFT coefficient separately in order to prevent the element coefficients with low discrimination power from being chosen as organizing attributes. Second, it selects the element coefficients with high discrimination power by performing the pre-processing step of analyzing the characteristics of a target database. As criteria for discrimination power, it uses the standard deviation. Third, using the cost model appropriate for similarity search, it estimates the costs of the index and object accesses, and determines the optimal number of organizing attributes systematically. To verify the effectiveness of the proposed method, we performed a series of experiments. The experimental results show that the proposed method outperforms the previous ones dramatically.

Acknowledgements

This research was partially supported by the 2002 Basic Research Program(Grant R05-2002-000-01085 -0) of the KOSEF and the 2001 University Research Program of the MIC in Korea. Sang-Wook Kim would like to thank Jung-Hee Seo, Suk-Yeon Hwang, Grace(Joo-Young) Kim, and Joo-Sung Kim for their warm encouragement and support.

References

[1] R. Agrawal, C. Faloutsos, and A. Swami, "Efficient similarity search in sequence databases", *Proc. Int'l. Conf. on Foundations of Data Organization and Algorithms (FODO)*, pages 69-84, 1993.

[2] R. Agrawal, K.-I. Lin, H. S. Sawhney, and K. Shim, "Fast similarity search in the presence of noise, scaling, and translation in time-series databases", *Proc. VLDB*, pages 490-501, 1995.

[3] A. Belussi and C. Faloutsos, "Estimating the selectivity of spatial queries using the correlation fractal dimension", *Proc. VLDB*, pages 299-310, 1995.

[4] S. Berchtold, D. A. Keim, and H.-P. Kriegel, "The X-tree: An index structure for high-dimensional data", *Proc. VLDB*, pages 28-39, 1996.

[5] C. Faloutsos and I. Kamel, "Beyond uniformity and independence: Analysis of R-trees using the concept of fractal dimension", *Proc. ACM SIGMOD*, pages 4-13, 1994.

[6] C. Faloutsos, M. Ranganatha, and Y. Manolopoulos, "Fast subsequence matching in time-series databases", *Proc. ACM SIGMOD*, pages 419-429, 1994.

[7] D. O. Goldin and P. C. Kanellakis, "On similarity queries for time-series data: Constraint specification and implementation", *Proc. Intl. Conf. On Principles and Practice of Constraint Programming*, 1995.

[8] S. W. Kim, S. Park, and W. W. Chu, "An index-based approach for similarity search supporting time warping in large sequence databases", *Proc. IEEE ICDE*, pages 607-614, 2001.

[9] S. Park, S. W. Kim, J. S. Cho, and S. Padmanabhan, "Prefix-querying: An approach for effective subsequence matching under time warping in sequence databases", *Proc. ACM CIKM*, pages 255-262, 2001.

[10] S. Park, W. W. Chu, J. Yoon, and C. Hsu, "Efficient searches for similar subsequences of different lengths in sequence databases", *Proc. IEEE ICDE*, pages 23-32, 2000.

[11] D. Rafiei and A. Mendelzon, "Similarity-based queries for time-series data", *Proc. ACM SIGMOD*, pages 13-24, 1997.