LETTER

Hybrid Lower-Dimensional Transformation for Similar Sequence Matching*

Yang-Sae MOON[†a)], ***Member and*** **Jinho KIM**[†], ***Nonmember***

SUMMARY Lower-dimensional transformations in similar sequence matching show different performance characteristics depending on the type of time-series data. In this paper we propose a hybrid approach that exploits multiple transformations at a time in a single hybrid index. This hybrid approach has advantages of exploiting the similar effect of using multiple transformations and reducing the index maintenance overhead. For this, we first propose a new notion of *hybrid lower-dimensional transformation* that extracts various features using different transformations. We next define the *hybrid distance* to compute the distance between the hybrid transformed points. We then formally prove that the hybrid approach performs similar sequence matching correctly. We also present the index building and similar sequence matching algorithms based on the hybrid transformation and distance. Experimental results show that our hybrid approach outperforms the single transformation-based approach.

key words: *databases, data mining, similar sequence matching, time-series data, lower-dimensional transformation*

1. Introduction

Time-series data are the sequences of real numbers representing values at specific points in time [1], [3], [5], [7], [8]. The time-series data stored in a database are called *data sequences*, and those given by users are called *query sequences*. Finding data sequences similar to the given query sequence from the database is called *similar sequence matching* [3], [9]. As the distance function $D(X, Y)$ between two sequences $X = \{x_0, \ldots, x_{n-1}\}$ and $Y = \{y_0, \ldots, y_{n-1}\}$, many similar sequence matching models have used the L_p-distance ($= \sqrt[p]{\sum_{i=0}^{n-1} |x_i - y_i|^p}$), including the Manhattan distance and the Euclidean distance [1]–[3], [9].

Most of the similar sequence matching solutions have used the *lower-dimensional transformation* to store high-dimensional sequences into a multidimensional index [1]–[3], [9], [11]. The lower-dimensional transformations, however, show different characteristics in indexing performance according to the type of time-series data. This means that there is no specific transformation showing the optimal performance for all types of time-series data, but the optimal transformation varies according to the type of time-series data. Based on this characteristics, Keogh et al. [6] proposed *E-Index*(Ensemble-Index) that used multiple indexes to exploit multiple lower-dimensional transformations. This approach is novel, but has two problems: 1) it has to know the characteristics of time-series data in advance, and 2) it incurs the index maintenance overhead due to use of multiple indexes.

In this paper we propose a hybrid approach that exploits multiple lower-dimensional transformations at a time in a multidimensional index. Motivation of the hybrid approach is based on the fact that the well-known transformations concentrate most of the energy into only a few features [1], [2], [8], [10]. Thus, our hybrid approach extracts only a few features from each transformation and integrates the features for a new lower-dimensional transformation. To do this, we first propose a new notion of *hybrid lower-dimensional transformation*(or simply *hybrid transformation*). For a given sequence, the hybrid transformation extracts various features with different characteristics by using different transformations and then integrates the features into a lower-dimensional point. We next define the *hybrid distance* to compute the distance between the hybrid transformed points. We then formally prove the correctness of the hybrid transformation-based similar sequence matching. We also present the index building and similar sequence matching algorithms based on the hybrid transformation and distance. Likewise, considering multiple transformations for all time-series we can solve the first problem of [6], and using multiple features in a single index we can also solve the second problem of [6]. Experimental results show that our hybrid approach outperforms the single transformation-based approach by taking superior characteristics of individual transformations.

2. Related Work

From now on, we use the following notations: F_i is the i-th lower-dimensional transformation; $F_i(S)$ is the transformed point from a sequence S by the transformation F_i; and $F_i(S)F_j(S)$ is a concatenation of $F_i(S)$ and $F_j(S)$.

Most of similar sequence matching solutions have used the *lower-dimensional transformation* to store high-dimensional sequences into a multidimensional index [1], [3], [5], [6], [11], since storing high-dimensional sequences in an index causes the high dimensionality problem and requires excessive index space [1], [3], [9]. Various transforms including DFT (Discrete Fourier Transform), Wavelet, and PAA (Piecewise Aggregate Approximation) are used as the lower-dimensional transformation of high-dimensional se-

Manuscript received May 23, 2008.
Manuscript revised October 16, 2008.

[†]The authors are with the Department of Computer Science, Kangwon National University, Korea.

*This study was supported by the Research Grant from Kangwon National University.

a) E-mail: ysmoon@kangwon.ac.kr

DOI: 10.1587/transinf.E92.D.541

quences. DFT is most widely used in many similar sequence matching solutions [1], [3], [9], [10]. Wavelet and PAA are also used in [2], [5] and [6], respectively. Besides these transforms, DCT (Discrete Cosine Transform) and SVD (Singluar Value Decomposition) are introduced as the lower-dimensional transformation [6]. Among these transformations, however, we cannot choose an optimal one since it varies by the type of time-series data [6].

Keogh et al. [6] proposed *E-Index* that used multiple indexes for multiple transformations. E-Index consists of multiple indexes, and each of which is used for a lower-dimensional transformation. They first select the most appropriate transformation for each part of time-series data, and then store the part of data in the corresponding index using the selected transformation. This approach is novel, but has the following two problems. First, it is difficult to select the most appropriate transformation in advance since the type of time-series data is too diverse, and the data will be updated and appended continuously. Second, E-Index incurs the index maintenance overhead since it needs to maintain multiple indexes internally. Therefore, in this paper we propose a hybrid approach that considers multiple transformations in a single index at a time.

3. Hybrid Lower-Dimensional Transformation

3.1 The Concept

We devise the hybrid transformation based on the fact that many transforms such as DFT and Wavelet concentrate most of the energy into only a few features. That is, we define a new lower-dimensional transformation by concatenating the features resulted from different lower-dimensional transformations.

Definition 1: Given m lower-dimensional transformations $F_1, F_2, \ldots, F_m$, the *hybrid lower-dimensional transformed* point of a sequence S, denoted by $HT(S)$, is defined as in Eq. (1):

$$HT(S) \equiv F_1(S)F_2(S)\cdots F_m(S) \tag{1}$$

Figure 1 depicts the hybrid transformation. As shown in the figure, we first transform the given sequence S using each transformation F_i, and then construct the hybrid transformed point $HT(S)$ by concatenating the energy concentrated features $F_i(S)$. We devise the hybrid transformation in order to exploit various characteristics of different transformations, since each transformation has its own characteristics [6]. Using the hybrid transformation we can get the similar effect of using multiple transformations in an index at a time.

To use the hybrid transformation in similar sequence matching, however, we need a new distance measure for the hybrid transformed sequences because the original L_p-distance will not satisfy Parseval's theorem [1] any more for the hybrid transformation. That is, the following Eq. (2) is not satisfied any more for a data sequence S and a query sequence Q.

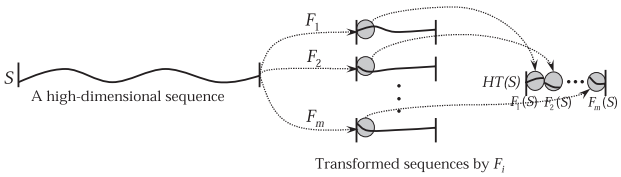


Fig. 1 Concept of the hybrid transformation.

$$D(S, Q) \le \epsilon \Rightarrow D(HT(S), HT(Q)) \le \epsilon \tag{2}$$

The lower-dimensional transformation in similar sequence matching should satisfy Eq. (2) [1], [3], [9]. Thus, if we want to use the hybrid transformation in similar sequence matching, we need to define a new distance measure for the hybrid transformed points.

Definition 2: If a data sequence S and a query sequence Q are transformed to $HT(S)$ and $HT(Q)$, respectively, by the hybrid transformation w.r.t. $F_1, F_2, \ldots, F_m$, then the *hybrid distance* between two transformed points $HT(S)$ and $HT(Q)$, denoted by $HD(HT(S), HT(Q))$, is defined as in Eq. (3):

$$HD(HT(S), HT(Q)) \equiv \max_{1 \le i \le m} \{D(F_i(S), F_i(Q))\} \tag{3}$$

In Definition 2, we select the maximum distance as the hybrid distance in order to use the tightest bound when we compare the transformed points. Since the larger distance provides the better indexing performance in similar sequence matching [3], [9], by using the maximum distance we can get the better performance in indexing the transformed points.

Theorem 1 shows that the hybrid transformation satisfies Parseval's theorem for the hybrid distance.

Theorem 1: Given a data sequence S and a query sequence Q, if the lower-dimensional transformations $F_1, F_2, \ldots, F_m$ satisfy Parseval's theorem, then the L_p-distance $D(S, Q)$ and the hybrid distance $HD(HT(S), HT(Q))$ satisfy the following Eq. (4):

$$D(S, Q) \le \epsilon \Longrightarrow HD(HT(S), HT(Q)) \le \epsilon \tag{4}$$

Proof: By Parseval's theorem, every F_i satisfies the relationship of $D(S, Q) \le \epsilon \Rightarrow D(F_i(S), F_i(Q)) \le \epsilon$ [1]. Thus, $D(S, Q) \le \epsilon \Rightarrow \max_{1 \le i \le m}\{D(F_i(S), F_i(Q))\} \le \epsilon$ also holds. By Definition 2, $\max_{1 \le i \le m}\{D(F_i(S), F_i(Q))\}$ is denoted as $HD(HT(Q), HT(S))$. It means that Eq. (4) holds. □

Theorem 1 means that we can perform similar sequence matching correctly if we use the hybrid distance for the hybrid transformation.

3.2 Index Building and Similar Sequence Matching Algorithms

To perform similar sequence matching, we need to construct a multidimensional index first. Index building mechanism

for the hybrid transformation is the same as that for the traditional transformation. That is, using the hybrid transformation we first transform high-dimensional sequences to low-dimensional points, and then construct an index by storing the transformed points. Algorithm 1 shows the index building algorithm that uses the hybrid transformation. Algorithm 1 is for *whole matching* [1], [2], and by referring [3], [9] it can be easily extended to that for *subsequence matching*. In Line 2, we transform each data sequence to a low-dimensional point using the hybrid transformation. After then, in Line 3 we store the transformed point into the index with a pointer to the corresponding data sequence. By repeating these two steps for every data sequence, we can complete the index construction (Lines 1 to 4). We use the index built by Algorithm 1 to obtain the *candidate* [1], [3], [9] data sequences in the following similar sequence matching algorithm.

Algorithm 1 *BuildIndex* $(\mathbb{S}, F_1, \ldots, F_m)$

Input: $\mathbb{S}$ is a set of data sequences; F_i is the i-th lower-dimensional transformation.
1: **for each** data sequence $S \in \mathbb{S}$ **do**
2: Obtain the hybrid transformed point $HT(S)$ using $F_1, \ldots, F_m$;
3: Insert $HT(S)$ into the index with a pointer to S;
4: **end for**

We now explain the similar sequence matching algorithm. For the traditional transformation, we generally use the L_p-distance in searching the index to compare two transformed points. In contrast, for the hybrid transformation, we need to use the hybrid distance of Definition 2 in searching the index. It is because, as we explained in Sect. 3.1, the traditional transformation satisfies Parseval's theorem, in contrast, the hybrid transformation satisfies Theorem 1, the modified Parseval's theorem for the hybrid transformation.

Algorithm 2 shows the whole matching algorithm [1], which can be extended to the subsequence matching algorithm by [3], [9], [11] and the k-nearest neighbor algorithm by [2], [5], [6]. As shown in Algorithm 2, we first construct a range query using the hybrid transformed point and the given tolerance (Lines 1 and 2). We then obtain the candidate data sequences by evaluating the range query on the index (Line 3). In this index searching step, we use the hybrid distance instead of the original L_p-distance since we use the hybrid transformation instead of the traditional transformation. After obtaining the candidate data sequences, we perform the *post-processing step* [1]–[3], [9] that identifies only true similar sequences by accessing the actual data sequences (Line 4). We finally return the true similar sequences as the results (Line 5).

Algorithm 2 *WholeMatching* $(Q, \epsilon, F_1, \ldots, F_m)$

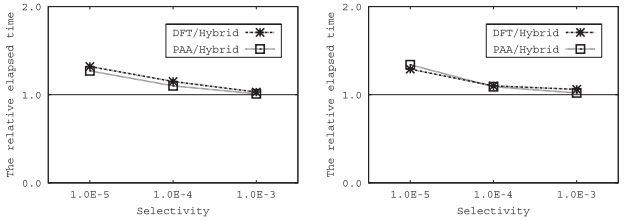
Input: Q is a query sequence; ϵ is the tolerance; F_i is the i-th lower-dimensional transformation.
1: Obtain the hybrid transformed point $HT(Q)$ using $F_1, \ldots, F_m$;
2: Construct a range query using $HT(Q)$ and ϵ;
3: Search the index using the hybrid distance;
4: Perform the post-processing step to remove false alarms;
5: Return the true similar data sequences;

4. Performance Evaluation

We performed experiments using four types of data sets: the first one is *STOCK-DATA* that contains a real stock data [3], [8], [9] consisting of 329,112 entries; the second one is *WALK-DATA* that contains random walk synthetic data [3], [9] consisting of one million entries; the third one is *SINE-DATA* that contains synthetic streaming time-series [4] consisting of one million entries; and the fourth one is *PERIOD-DATA* that contains pseudo periodic synthetic time-series [9] consisting of one million entries. For lower-dimensional transformations, we use two transforms, DFT and PAA, since the characteristics of DCT and Wavelet are similar to those of DFT and Wavelet, respectively [6]. We use both DFT and PAA for the hybrid transformation and extract four features from each transformation. As the similar sequence matching method, we use Dual Match [9], a subsequence matching method, and as the multidimensional index, we use the R*-tree [3]. For the experimental results, we measure the elapsed time of each transformation because the filtering efficiency of transformations can be measured as the query response time, and most papers [3], [6], [9] used the response time as the efficient measure of transformations. We set the query sequence length to 256, but change the selectivity [3], [9] from 10^{-5} to 10^{-3}. To avoid effects of noise, we experiment with ten different query sequences and use the average as the result. The hardware platform is a PC equipped with an Intel Pentium IV 2.80 GHz CPU and 512 MB RAM. The software platform is GNU/Linux Version 2.6.6 operating system.

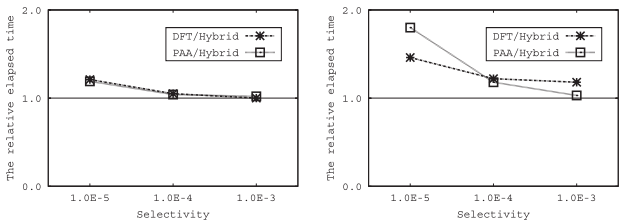
Figure 2 shows the experimental results for STOCK-DATA and WALK-DATA. In the graphs, the horizontal axis represents the selectivity, and the vertical axis the relative elapsed time between DFT (or PAA) and the hybrid transformation. As shown in Fig. 2 (a), DFT and PAA show the similar performance trend, but the hybrid transformation is always superior to DFT and PAA. It is because, for each part of time-series data, we select the best transformation and use it as the hybrid transformation. In Fig. 2 (a), the performance difference decreases as the selectivity increases. It is because, as the selectivity increases, the time for the post-processing step becomes much longer than that for the index searching step. The result for WALK-DATA in Fig. 2 (b) is very similar to that for STOCK-DATA, since WALK-DATA has the similar characteristics with STOCK-DATA [3].

Figure 3 shows the results for SINE- and PERIOD-DATA. We note that the overall trend is similar to that of Fig. 2, and the hybrid transformation also shows the better performance than DFT and PAA. Especially in Fig. 3 (b), DFT is better than PAA for low selectivity ranges, but PAA is better than DFT for high selectivity ranges. Likewise, the best transformation can vary by the selectivity range. Our hybrid transformation, however, shows the best performance in all the selectivity ranges since it integrates both DFT and PAA. In summary, the hybrid transformation shows the better performance than the traditional transformations for all



(a) STOCK-DATA. (b) WALK-DATA.

Fig. 2 Experimental results on STOCK- and WALK-DATA.



(a) SINE-DATA. (b) PERIOD-DATA.

Fig. 3 Experimental results on SINE- and PERIOD-DATA.

types of time-series data and for all ranges of selectivity. It means that we can use the hybrid transformation without considering the data characteristics and the selectivity ranges.

5. Conclusions

The previous approach [6] with multiple transformations has a problem of requiring the investigation on time-series characteristics in advance and another problem of incurring the index maintenance overhead. To solve these problems, we proposed a hybrid lower-dimensional transformation and presented the similar sequence matching method based on that hybrid transformation. Using the hybrid transformation we can exploit the similar effect of using multiple lower-dimensional transformations in an index at a time. We also reduced the index maintenance overhead by using only one hybrid index. Through experiments for various data sets, we also showed that our hybrid approach outperformed the single transformation-based approach. These results indicate that our hybrid transformation can be widely used for various time-series data with different characteristics. Future research includes extension of the hybrid transformation to support the dynamic time warping distance [5], [7]

References

[1] R. Agrawal, C. Faloutsos, and A. Swami, "Efficient similarity search in sequence databases," Proc. 4th Int'l Conf. on Foundations of Data Organization and Algorithms, pp.69–84, Oct. 1993.
[2] K.-P. Chan, A.W.-C. Fu, and C.T. Yu, "Haar wavelets for efficient similarity search of time-series: With and without time warping," IEEE Trans. Knowl. Data Eng., vol.15, no.3, pp.686–705, Jan./Feb. 2003.
[3] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast subsequence matching in time-series databases," Proc. Int'l Conf. on Management of Data, pp.419–429, ACM SIGMOD, May 1994.
[4] L. Gao and X.S. Wang, "Continually evaluating similarity-based pattern queries on a streaming time series," Proc. Int'l Conf. on Management of Data, pp.370–381, ACM SIGMOD, June 2002.
[5] W.-S. Han, J. Lee, Y.-S. Moon, and H. Jiang, "Fast ranked subsequence matching in time-series databases," Proc. Int'l Conf. on Very Large Data Bases, pp.423–434, Vienna, Austria, Sept. 2007.
[6] E.J. Keogh, S. Chu, and M.J. Pazzani, "Ensemble-index: A new approach to indexing large databases," Proc. 7th Int'l Conf. on Knowledge Discovery and Data Mining, pp.117–125, ACM SIGKDD, Aug. 2001.
[7] E.J. Keogh, L. Wei, X. Xi, S.-H. Lee, and M. Vlachos, "LB_Keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures," Proc. Int'l Conf. on Very Large Data Bases, pp.882–893, Sept. 2006.
[8] H.-S. Lim, K.-Y. Whang, and Y.-S. Moon, "Similar sequence matching supporting variable-length and variable tolerance continuous queries on time-series data stream," Inf. Sci., vol.178, no.6, pp.1461–1478, March 2008.
[9] Y.-S. Moon, K.-Y. Whang, and W.-S. Han, "General match: A subsequence matching method in time-series databases based on generalized windows," Proc. Int'l Conf. on Management of Data, pp.382–393, ACM SIGMOD, June 2002.
[10] Y.-S. Moon, "An MBR-safe transform for high-dimensional MBRs in similar sequence matching," Proc. 12th Int'l Conf. on Database Systems for Advanced Applications (DASFAA 2007), pp.79–90, Bangkok, Thailand, April 2007.
[11] Y.-S. Moon and J. Kim, "Efficient moving average transform-based subsequence matching algorithms in time-series databases," Inf. Sci., vol.177, no.23, pp.5415–5431, Dec. 2007.