Time Series Indexing By Dynamic Covering with Cross-Range Constraints*

Tao Sun §, Hongbo Liu †, Seán McLoone ‡, Shaoxiong Ji ♮, Xindong Wu ♯

§School of Innovation and Entrepreneurship, Dalian University of Technology, China

†Institute of Cognitive Information Technology, Dalian Maritime University, China

‡School of Electronics, Electrical Engineering and Computer Science at Queen's University Belfast, UK

♮Department of Computer Science, Aalto University, Finland

♯Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, China; and Mininglamp Academy of Sciences, Mininglamp Technology, China

Emails: dlutst@dlut.edu.cn; lhb@dlmu.edu.cn; s.mcloone@qub.ac.uk; shaoxiong.ji@aalto.fi; xwu@hfut.edu.cn

Abstract

Time series indexing plays an important role in querying and pattern mining of big data. This paper proposes a novel structure for tightly covering a given set of time series under the dynamic time warping similarity measurement. The structure, referred to as Dynamic Covering with cross-Range Constraints (DCRC), enables more efficient and scalable indexing to be developed than current hypercube based partitioning approaches. In particular, a lower bound of the DTW distance from a given query time series to a DCRC-based cover set is introduced. By virtue of its tightness, which is proven theoretically, the lower bound can be used for pruning when querying on an indexing tree. If the DCRC based Lower Bound (LB_DCRC) of an upper node in an index tree is larger than a given threshold, all child nodes can be pruned yielding a significant reduction in computational time. A Hierarchical DCRC (HDCRC) structure is proposed to generate the DCRC-tree based indexing and used to develop time series indexing and insertion algorithms. Experimental results for a selection of benchmark time series datasets are presented to illustrate the tightness of LB_DCRC, as well as the pruning efficiency on the DCRC-tree, especially when the time series have large deformations.

Keywords— Time Series; Dynamic Time Warping; Indexing; R-Tree; Dynamic Covering; Cross-Range Constraints

1 Introduction

With the dramatic growth in the volume of data, and the opportunities for data driven decision making afforded by such data, particularly when it comes to social networks and e-commerce [18, 40], it is vital to have algorithms that are able to efficiently mine big data [2, 36]. In many practical applications mining of data that is in the form of time series [5, 10] is of interest and this has led to the development of bespoke approaches for tasks such as pattern discovery and clustering [37, 21, 9], classification [7, 20], rule discovery [30, 34], and summarisation [13]. As with standard data mining, indexing is a fundamental technique for efficiently accessing and querying data when performing these tasks [6, 4]. However, when indexing time series data the choice of similarity measurement is a key consideration [23], particularly when they are not aligned temporally. In these circumstances, the classical Euclidean distance, as introduced in [1], can result in large differences between two time series even when they are quite similar in shape [14]. Consequently, dynamic time warping (DTW), which addresses this deficiency, has become a popular method of measuring the similarity between time series [25, 22, 35, 24].

When indexing big time series datasets performing a direct linear scan of all the time series is generally computationally intractable and a more considered approach is needed. This usually involves mapping the data to a tree-like structure with partitions, and then extracting a small number of time series from these partitions for linear scanning [26, 39]. A partition is defined as a low-complexity structure covering a set of relatively similar time series. For a given query time series, a lower bound with respect to each partition can then be employed during indexing instead of directly measuring the similarity between the query time series and each element of the partitions. Using this approach efficient pruning procedures can be implemented, substantially reducing the computational complexity of indexing, and enabling fast data access and querying [14]. The speed-ups achievable using time series partitioning very much depend on how the partitions are defined, the approach used to generate tree-like indexing using these partitions, and the complexity of the lower bound calculation, hence improving on each of these remains an important area of research, and is the focus of this paper.

*Published in The VLDB Journal

In the classical methods [14, 39, 15], when computing the lower bound of DTW from a query time series $\mathbf{q}$ to a set S of time series, the range $[L_i, U_i]$ is computed for each dimension i. The set of dimensional ranges $[L_i, U_i]$, $i = 1, \cdots, m$, define a hyper rectangular area, denoted by C, which can serve as a partition in an indexing structure. In fact, the lower bound of DTW is exactly the Hausdorff distance from $\mathbf{q}$ to C. However, a partition represented by a hyper rectangle is often not optimal in terms of DTW distance. With the deformation of the time axis when DTW matching, the volume of partition C can be so large that C might still include quite dissimilar time series, even if the elements of S are similar, which results in inefficient indexing.

As an alternative to hyper rectangles, we propose the use of Dynamic Covering with Cross-Range Constraints (DCRC) to partition time series for indexing.

For a given set S, let an approximately central element in terms of the DTW distance be the "reference" time series, denoted $\mathbf{c}$. DCRC is defined as a series of sets $V_1, V_2, \cdots, V_m$. Each element of set V_i is a 3-tuple (l, u, p), where p is a dimensional subscript of reference $\mathbf{c}$, and $[l, u]$ denotes a dimensional range. A tuple $(\mathbf{v}_1, \mathbf{v}_2, \cdots)$ over the Cartesian product $V_1 \times V_2 \times \cdots$ corresponds to an m-dimensional hyper rectangle. We only consider tuples satisfying "Alignment", "Continuity" and "Monotonicity" conditions, called the ACM-relationship. These tuples under the ACM-relationship correspond to multiple m-dimensional hyper rectangles.

In contrast to the classic method, DCRC proposes a "tight" structure composed of multiple hyper rectangles. For a given set S of similar time series in terms of DTW distance, any element of the corresponding DCRC must be similar to the elements of S. The tightness makes it possible to efficiently prune unnecessary samples when partitioning for DTW indexing.

We determine the lower bound of the DTW between a given query time series and the cover set of a given DCRC structure, denoted as LB_DCRC, and then introduce the hierarchical DCRC (HDCRC) structure. This is composed of multiple layers, with the upper DCRC structure covering all the elements covered by the DCRC structures of its sub-layers. Based on the DCRC and HDCRC structures, we further present a novel tree-like indexing and its insertion and node splitting algorithms. Given time series set S and a query time series $\mathbf{q}$, from the root down to its sub-layers in the indexing tree, if the LB_DCRC (DCRC based Lower Bound of DTW) of an upper layer is larger than a given acceptable range query tolerance, then all of its sub-layers are accordingly pruned, with the result that only a few remaining leaves on the indexing tree need to be sequentially scanned using the DTW distance. This leads to significant reductions in computational time.

In summary, the novel contributions of the paper are as follows:

(a) We develop the theory of DCRC-based covering of a given set of time series, and prove that a DCRC-based covering has significantly lower volume than other methods, that is, if all the elements are similar to the reference $\mathbf{c}$, any element of the corresponding DCRC-based cover set is also similar to $\mathbf{c}$.

(b) The corresponding lower bound of the DTW between a given query time series and a given time series set, namely, LB_DCRC is proposed. This bound outperforms other lower bounds in terms of tightness.

(c) Since the number of feasible ACM-relationships for a given DCRC usually grows exponentially, we propose a novel polynomial time algorithm to compute the lower bound of the DTW between a given query time series and the cover set of a given DCRC structure.

(d) We then present the hierarchical DCRC (HDCRC) structure, HDCRC-based tree indexing and its insertion and node splitting algorithms and demonstrate with extensive numerical studies that the proposed DCRC based indexing method performs efficient pruning for range querying, and outperforms linear scanning and other indexing methods in terms of computational time.

The remainder of the paper is organized as follows. Related work is reviewed in section 2. The key DCRC concepts and alrorithms are introduced in section 3. Then the HDCRC structure and the indexing approach based on the DCRC-tree are developed in section 4. The relevant theorems on DCRC and HDCRC are presented in section 5. Using benchmark datasets from the UCR Time Series Classification Archive, experimental results are provided in section 6 to demonstrate the efficiency of our approaches. Finally, conclusions are provided in section 7.

2 Related Work

DTW is a more robust measure of the similarity between two time series than the Euclidean distance as it takes account of time axis shifting between time series. Generally, the warping path of DTW is defined by a number of global and/or local constraints. Two of the most popular global constraints are the Itakura parallelogram [12] and the Sakoe-Chiba band [28]. In contrast to the traditional form of DTW, this paper adopts the form DTW_p [16, 32] to denote the L_p norm of monotonic DTW distance ($p = 2$).

Despite its limitation with respect to scalability to high dimensional data sets, in recent years DTW has been widely applied, particularly for high-dimensional data indexing [33] and stream matching [19, 11].

However, since DTW does not obey the triangle inequality, and therefore is not suitable for indexing with a metric access method, researchers have switched their attention to developing indexing approaches that work with suitability defined DTW lower bounds, rather than DTW itself. In recent years, many researches have focused on the DTW lower bound.

The idea of using a lower bound function was first proposed by Yi et al. [38]. In their lower bound, denoted as LB_Yi, the maximum and minimum elements of a sequence are used to represent the sequence.

Keogh et al. proposed a lower bound function (denoted as LB_Keogh) [14], together with an exact indexing method based on their lower bound function. For two given time series $\mathbf{x}$ and $\mathbf{y}$, let Y be a range series, each entry Y_i of which denotes the i-th envelope, i.e. the range between the minimum and the maximum of the warping window with center y_i. In fact, LB_Keogh corresponds to the Hausdorff distance from $\mathbf{x}$ to Y.

Lemire proposed LB_IMPROVED lower bound [16], which imports additional time series $\mathbf{x}'$ from $\mathbf{x}$ and Y, and the lower bound is represented by LB_Keogh($\mathbf{x}$,$\mathbf{y}$) + LB_Keogh($\mathbf{y}$,$\mathbf{x}'$).

Based on the common features of LB_Kim, LB_Yi and LB_Keogh, Zhou and Wong [39] proposed several boundary-based lower bound functions including a non-elaborate version (denoted as LB_Corner) and an elaborate version (denoted as LB_ECorner). Li and Yang [17] proposed two extensions of LB_Kim and LB_Keogh (denoted respectively as LB_NKim and LB_NKeogh).

In 2018 Shen et al. proposed a new lower bound (LB_NEW) [29]. In contrast to LB_Keogh, LB_NEW defines Y_i as all the elements of the warping window with center y_i, instead of the i-th envelope Y_i in LB_KEOGH. Therefore, LB_NEW is usually tighter than LB_Keogh. Tan et al. [32] proposed the LB_ENHANCED lower bound. In this algorithm, Y_i is represented by left bands $\mathcal{L}_i^W$ or right bands $\mathcal{R}_i^W$, assuring a relatively tight lower bound.

In the traditional time series indexing methods [14], the dataset S of sample time series is stored in an R-tree like structure, each tree node of which corresponds to a minimal boundary rectangle (MBR) containing a subset of S. Given a query time series $\mathbf{q}$, retrieving the subset $\{\mathbf{s} \in S | DTW(\mathbf{q}, \mathbf{s}) \leq \varepsilon\}$ involves two steps:

(1) Seach the nodes based on the lower bound between $\mathbf{q}$ and MBR in a top-down approach.

(2) All the feasible time series are linear scanned using an efficient method [27].

3 Dynamic Covering with Cross-Range Constraints (DCRC)

3.1 DTW

Given a time series $\mathbf{x}$ represented by $[x_1, x_2, \cdots, x_n]$, let $\mathbf{x}(i)$ denote the i-th entry of $\mathbf{x}$, x_i and $\mathbf{x}(i_1 : i_2)$ denote the subsequence $[x_{i_1}, x_{i_1+1}, \cdots, x_{i_2}]$. Here, n is the length of the time series, also referred to as its "dimension".

DTW measures the similarity between two time series [31]. For two given time series $\mathbf{x} = [x_1, x_2, \cdots, x_m]$ and time series $\mathbf{y} = [y_1, y_2, \cdots, y_n]$, let $\mathbf{W}$ denote a warping path from $\mathbf{x}$ to $\mathbf{y}$. Let (i_k, j_k) be the k-th element of $\mathbf{W}$ and K be the length of $\mathbf{W}$ $(1 \leq k \leq K)$. The warping path in DTW is required to satisfy a set of constraints, referred to as alignment, continuity and monotonicity constraints. These are defined as follows:

(a) $(i_1, j_1) = (1, 1)$ and $(i_K, j_K) = (m, n)$;

(b) $i_{k+1} - i_k \leq 1$ and $j_{k+1} - j_k \leq 1$, $k = 1, 2, \cdots, K-1$;

(c) $i_{k+1} - i_k \geq 0$ and $j_{k+1} - j_k \geq 0$, $k = 1, 2, \cdots, K-1$.

The ratio of the width of the Sakoe-Chiba Band to the length of the time series, denoted by λ $(0 < \lambda \leq 1)$, imposes an additional constraint which is defined as follows:

(d) $|\frac{n}{m} i_k - j_k| \leq \lambda n$, $k = 1, 2, \cdots, K$.

The DTW path distance is obtained subject to these constraints by solving the dynamic programming problem given in Equ. (1), where $\delta(i,j) = (x_i - y_j)^2$, $\sqrt{\mu(i,j)}$ represents the DTW distance between $\mathbf{x}(1:i)$ and $\mathbf{y}(1:j)$, and $DTW(\mathbf{x}, \mathbf{y}) = \sqrt{\mu(m,n)}$.

$$\mu(i,j) = min \begin{cases} \delta(i,j) + \mu(i-1, j-1) \\ \delta(i,j) + \mu(i-1, j) \\ \delta(i,j) + \mu(i, j-1) \end{cases} \tag{1}$$

3.2 ACM-Relationship

Definition 1 (ACM-Relationship) *Considering the Cartesian product $P_1 \times P_2 \times \cdots \times P_m$, where $P_i = \{1, 2, \cdots, n\}$ for $i = 1, 2, \cdots, m$. Let $\mathbb{R}(m,n)$ denote the relationship on the Cartesian product, each element $\mathbf{r}[r_1, r_2, \cdots, r_m]$ of which satisfies the Alignment, Continuity and Monotonicity (ACM-Relationships) as follows.*

(a) Alignment. $r_1 = 1, r_m = n$;

(b) Continuity. $r_{i+1} - r_i \leq 1$ for $i = 1, 2, \cdots, m-1$;

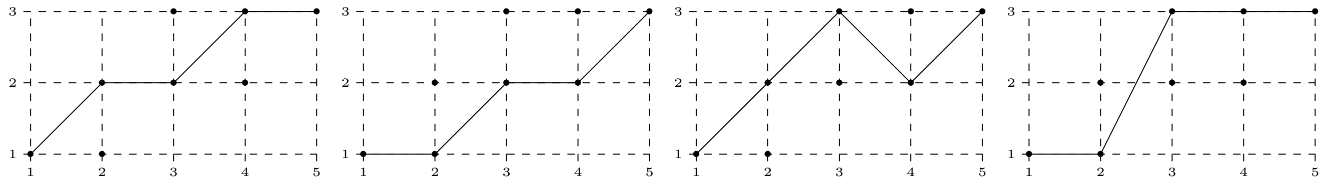
(c) Monotonicity. $r_{i+1} - r_i \geq 0$ for $i = 1, 2, \cdots, m-1$.

Given a time series $\mathbf{x}[x_1, x_2, \cdots, x_n]$ of length n, and a relationship $\mathbf{r}[r_1, r_2, \cdots, r_m] \in \mathbb{R}(m,n)$, let

$$\tau(\mathbf{x}, \mathbf{r}) = [x_{r_1}, x_{r_2}, \cdots, x_{r_m}] \tag{2}$$

Given a time series $\mathbf{x}[x_1, x_2, \cdots, x_m]$ of length m, and a time series $\mathbf{y}[y_1, y_2, \cdots, y_n]$ of length n, let

$$\begin{cases} \mathcal{R}(\mathbf{x}, \mathbf{y}) = \underset{\mathbf{r} \in \mathbb{R}(m,n)}{argmin} ||\mathbf{x}, \tau(\mathbf{y}, \mathbf{r})|| \\ \mathcal{D}(\mathbf{x}, \mathbf{y}) = \underset{\mathbf{r} \in \mathbb{R}(m,n)}{min} ||\mathbf{x}, \tau(\mathbf{y}, \mathbf{r})|| \end{cases} \tag{3}$$



(a) A legal ACM-Relationship (b) A legal ACM-Relationship (c) An illegal ACM-Relationship as it violates "Monotonicity" (d) An illegal ACM-Relationship as it violates "Continuity"

Figure 1: Examples of legal and illegal ACM-relationships

In Equ. (3), $\mathbf{r}$ is a ACM-Relationship, $\tau(\mathbf{y}, \mathbf{r})$ is a time series of length m while $|\mathbf{y}| = n < m$, and $||\mathbf{x}, \tau(\mathbf{y}, \mathbf{r})||$ is the Euclidian distance of the two m-length time series $\mathbf{x}$ and $\tau(\mathbf{y}, \mathbf{r})$. $\mathcal{D}(\mathbf{x}, \mathbf{y})$ is the minimum Euclidian distance with respect to relationship $\mathbf{r}$, and $\mathcal{R}(\mathbf{x}, \mathbf{y})$ is the corresponding value of $\mathbf{r}$.

Fig. 1 shows examples of the ACM-relationship. In each sub-figure of Fig. 1, 5 columns correspond to 5 sets $P_1, P_2, \cdots, P_5$, and the black dots correspond to the elements of P_i. The black dots on the black path represent elements of the Cartesian product $P_1 \times P_2 \times \cdots \times P_5$. The two series represented by Figs. 1(a) and 1(b) satisfy the ACM-relationships. However, the two series in Figs. 1(c) and 1(d) do not satisfy the ACM-relationships.

Algorithm 1 Minimization for ACM-Relationship

Input: A given time series $\mathbf{x}[x_1, x_2, \cdots, x_m]$ of length m, and a time series $\mathbf{y}[y_1, y_2, \cdots, y_n]$ of length $n (n < m)$.
Output: $\mathbf{r}[r_1, r_2, \cdots, r_m] = \mathcal{R}(\mathbf{x}, \mathbf{y})$ and $d = \mathcal{D}(\mathbf{x}, \mathbf{y})$.

```
Let μ_00 = 0, let μ_i0 = ∞ for i = 1, 2, ···, m, and let μ_0j = ∞ for j = 1, 2, ···, n;
for i = 1 to m, j = 1 to n do
    Let p = argmin_{q∈{j−1,j}} δ(i − 1, q);
    Let r_{i−1} = p;
    Let μ_ij = δ(i, j) + μ_{i−1,p};
end for
Let r_m = n;
return r = [r_1, r_2, ···, r_m], and d = √μ_mn;
```

3.3 Approximate Subsequence

Let $\mathcal{A}(i_1 : i_2)$ denote the mean of the entries of $\mathbf{x}(i_1 : i_2)$ and let $\mathcal{E}(i_1 : i_2)$ denote the sum of squares of deviations from the mean of the entries of $\mathbf{x}(i_1 : i_2)$ as defined in Equ. (4).

$$\begin{cases} \mathcal{A}(i_1 : i_2) = \frac{\sum_{j=i_1}^{i_2} x_j}{i_2 - i_1 + 1} \\ \mathcal{E}(i_1 : i_2) = \sum_{j=i_1}^{i_2} (x_j - \mathcal{A}(i_1 : i_2))^2 \end{cases} \tag{4}$$

Definition 2 (Approximate Subsequence) *For a given m-length time series $\mathbf{x}$ and a given integer n $(0 < n < m)$, the n-length Approximate Subsequence of $\mathbf{x}$, denoted by $\mathcal{AS}(\mathbf{x}, n)$ is defined as*

$$\mathcal{AS}(\mathbf{x}, n) = \underset{|\mathbf{y}|=n}{argmin}\, \mathcal{D}(\mathbf{x}, \mathbf{y}) \tag{5}$$

From Definition 2, the approximate subsequence of $\mathbf{x}$ is the approximate time series of $\mathbf{x}$. The optimal solution to Equ. (5), and hence $\mathcal{AS}(\mathbf{x}, n)$, is obtained by solving the dynamic program:

$$\nu(i, j) = \min_{k} \left(\nu(k - 1, j - 1) + \mathcal{E}(k : i)\right) \tag{6}$$

where $k \in \{j, j+1, \cdots, i\}$ and $\nu(i, j) = \mathcal{D}^2(\mathbf{x}(1 : i), \mathcal{AS}(\mathbf{x}(1 : i), j))$. The procedure for computing $\mathcal{AS}(\mathbf{x}, n)$ is given in Algorithm 2.

3.4 Covering Set

Consider a given set of m-length time series $S = \{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_{|S|}\}$, where $\mathbf{s}_i$ consists of $[s_{i1}, s_{i2}, \cdots, s_{im}]$. In this section, we focus on defining a structure that can tightly cover set S using the DTW distance.

Algorithm 2 Approximate subsequence for a given time series

Input: m-length time series $\mathbf{x}$.
Output: n-length Approximate subsequence.

Initialise an n-length time series $\mathbf{y}$;
Let $i = m$;
for $j = n$ down to 1 **do**
  Let $p = i$;
  Let $i = \underset{k}{argmin}\ (\nu(k-1, j-1) + \mathcal{E}(k:i))$;
  Let $\mathbf{y}(j) = \mathcal{A}(i:p)$;
  Let $i = i - 1$;
end for
return $\mathbf{y}$

Given a positive integer n ($n < m$), we define a new structure $\mathbb{V}$ to store different dimensional ranges. Assume $\mathbb{V} = [V_1, V_2, \cdots, V_m]$, where each element $\mathbf{v}$ of V_i is represented by $\mathbf{v}(p, l, u)$. The component $p \in \{1, 2, \cdots, n\}$ denotes a dimensional subscript, and $[l, u]$ denotes an interval on the real line of the p-th dimension. We stipulate that for any given $\mathbf{v}_1, \mathbf{v}_2 \in V_i$, $\mathbf{v}_1.p = \mathbf{v}_2.p$ if and only if $\mathbf{v}_1 = \mathbf{v}_2$.

Fig. 2 shows an example of the structure. As shown in Fig. 2(a), the structure is composed of 5 sets $V_1, V_2, \cdots, V_5$, with each set containing a number of 3-tuples. Take V_1 for example in Fig. 2(b). There are two rectangles representing the two 3-tuples. The upper edge and the lower edge of each rectangle denote the range $[\mathbf{v}.l, \mathbf{v}.u]$, and the number in the rectangle denotes a subscript of the reference time series.

For the sake of convenience, we introduce the following notation.

$$\begin{cases} \mathbb{V}.n = max\{\mathbf{v}.p | \mathbf{v} \in V_m\} \\ \mathbb{V}.P_i = \{\mathbf{v}.p | \mathbf{v} \in V_i\} \\ \mathbb{V}.\mathbf{v}_i^j = \mathbf{v}(p, l, u) \ \ s.t.\ (\mathbf{v} \in V_i \wedge \mathbf{v}.p = j) \\ \mathbb{V}.L_i^j = \mathbb{V}.\mathbf{v}_i^j.l \\ \mathbb{V}.U_i^j = \mathbb{V}.\mathbf{v}_i^j.u \end{cases} \tag{7}$$

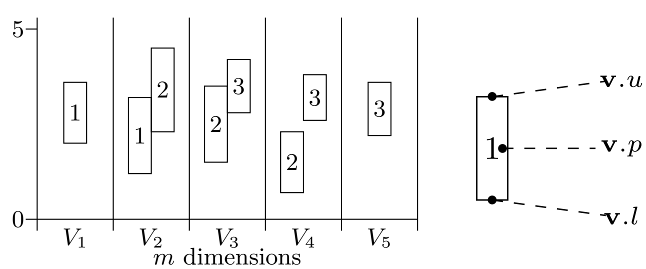
For a given $\mathbf{r} \in \mathbb{R}(m, n)$, let $Rect_r(\mathbb{V}, \mathbf{r})$, as defined in Equ. (8), be an m-dimensional hyper rectangular range.

$$Rect_r(\mathbb{V}, \mathbf{r}) = \{[x_1, \cdots, x_m] \mid x_i \in [L_i^j, U_i^j]\} \tag{8}$$

Fig. 3 illustrates the set in a hyper dimensional rectangle defined in Equ. (8). The first row denotes a matching path of DTW, the second row illustrates a DCRC structure, and the third row illustrates a 5-dimensional hyper rectangle. The lower and upper edges of each rectangle denote the corresponding range of each dimension. Take the third column for example. The value of the first row is 2, and then in the second row, the rectangle with label 2 is selected as the range corresponding to the third row.

$Rect_r(\mathbb{V}, \mathbf{r})$ corresponds to an m-dimensional cube for a given tuple $\mathbf{r}$, which covers a set of time series. In fact, not all tuples are permitted; a "legal" tuple $\mathbf{r}$ must obey the so-called "ACM"-Relationships.

$$volume(\mathbb{V}) = \prod_{1}^{n} (max\{\mathbf{v}.u - \mathbf{v}.l | \mathbf{v}.p = j \wedge \mathbf{v} \in V \in \mathbb{V}\}) \tag{9}$$



(a) (b)

Figure 2: Illustration of a DCRC structure: (a) A DCRC structure with 5 tuple sets; (b) A tuple $\mathbf{v}(p, l, u)$ in set V_1.

The structure $\mathbb{V}$ stores different dimensional ranges from the given set of time series, from which we can dynamically obtain a "legal" and "tight" cover of the given set. The "$Cover$" function is defined by Equ. (10).

$$Cover(\mathbb{V}) = \{\mathbf{x} \mid \mathbf{x} \in Rect_r(\mathbb{V}, \mathbf{r}), \mathbf{r} \in \mathbb{R}(m, n)\} \tag{10}$$

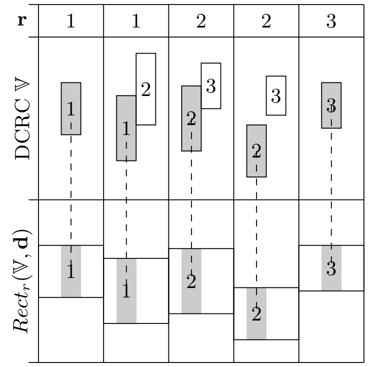


Figure 3: Illustration of hyper rectangle $Rect_r$ for 5-dimensional DCRC $\mathbb{V}$

where $Cover$ is a dynamic combination of $Rect_r(\mathbb{V}, \mathbf{r})$, where $\mathbf{r}$ is subject to the ACM-relationship. Hence, we refer to the covering structure as the "Dynamic Covering with Cross-Range Constraints" (DCRC for short).

3.5 DCRC of Time Series

In this section, a feasible and optimal algorithm for computing the DCRC for a given general set of time series is proposed. The required steps are set out in detail in Algorithm 3.

For a given set S of similar time series, the number of possible values that can be assigned to a DCRC structure grows exponentially with the number of their dimension. In our method, the creation of DCRC depends on a so-called reference $\mathbf{c}$ time series, which is understood to be a lower dimensional contour of all the samples of S. The ACM-Relationship $\mathbf{r}_t$ is just a many-to-one function from $\mathbf{s}_t$ to $\mathbf{c}$. In fact, the greater the similarity between the reference and the samples, the tighter the DCRC structure. The relevant theory is established in Theorem 3.

At Line 1, $\mathbf{c}$ is the reference time series for set S. To simply the computation, $\mathbf{c}$ is assigned to the n-length "Approximate Subsequence" of $\mathbf{s}_k$ randomly selected from set S. At Line 4, $\mathbf{r}$ is an ACM-Relationship by Algorithm 1. For each dimension i, the tuple set V_i of $\mathbb{V}$ is created or updated by the steps at Lines 5-13.

Table 1 illustrates a sample DCRC structure building procedure. $\mathbf{r}_t$ corresponds to the matching from $\mathbf{s}_t$ to $\mathbf{c}$ satisfying Equ. (3). X_i is the set of matchings (r_{ti}, s_{ti}). Y_i represents the merged set $\{(r, G_r)\}$ of X_i such that $r \in \{r_{ti}\}$ and $G_r = \{\mathbf{s}|(r, \mathbf{s}) \in X_i\}$, and V_i denotes the i-th entry of the DCRC structure.

Table 1: Example of computing a DCRC structure from two 5-length time series and a 3-length reference time series

	1	2	3	4	5
$\mathbf{c}$	4.00	6.00	5.00		
$\mathbf{s}_1$	4.11	4.12	4.13	6.14	5.15
$\mathbf{s}_2$	4.21	6.22	6.23	5.24	5.25
$\mathbf{s}_3$	4.31	6.32	6.33	6.34	5.35
$\mathbf{r}_1$	$\mathbf{1}(s_{11} \to c_1)$	$\mathbf{1}(s_{12} \to c_1)$	$\mathbf{1}(s_{13} \to c_1)$	$\mathbf{2}(s_{14} \to c_2)$	$\mathbf{3}(s_{15} \to c_3)$
$\mathbf{r}_2$	$\mathbf{1}(s_{21} \to c_1)$	$\mathbf{2}(s_{22} \to c_2)$	$\mathbf{2}(s_{23} \to c_2)$	$\mathbf{3}(s_{24} \to c_3)$	$\mathbf{3}(s_{25} \to c_3)$
$\mathbf{r}_3$	$\mathbf{1}(s_{31} \to c_1)$	$\mathbf{2}(s_{32} \to c_2)$	$\mathbf{2}(s_{33} \to c_2)$	$\mathbf{2}(s_{34} \to c_2)$	$\mathbf{3}(s_{35} \to c_3)$
$[X_i]$	$\{(\mathbf{1},s_{11}),$ $(\mathbf{1},s_{21}),$ $(\mathbf{1},s_{31})\}$	$\{(\mathbf{1},s_{12}),$ $(\mathbf{2},s_{22}),$ $(\mathbf{2},s_{32})\}$	$\{(\mathbf{1},s_{13}),$ $(\mathbf{2},s_{23}),$ $(\mathbf{2},s_{33})\}$	$\{(\mathbf{2},s_{14}),$ $(\mathbf{3},s_{24}),$ $(\mathbf{2},s_{34})\}$	$\{(\mathbf{3},s_{15}),$ $(\mathbf{3},s_{25}),$ $(\mathbf{3},s_{35})\}$
$[Y_i]$	$\{Y_1^1\{s_{11},s_{21},s_{31}\}\}$	$\{Y_2^1\{s_{12}\},$ $Y_2^2\{s_{22},s_{32}\}\}$	$\{Y_3^1\{s_{13}\},$ $Y_3^2\{s_{23},s_{33}\}\}$	$\{Y_4^2\{s_{14},s_{34}\},$ $Y_4^3\{s_{24}\}\}$	$\{Y_5^3\{s_{15},s_{25},s_{35}\}\}$
$[V_i]$	$\{(\mathbf{1},minY_1^1,maxY_1^1)\}$	$\{(\mathbf{1},minY_2^1,maxY_2^1),$ $(\mathbf{2},minY_2^2,maxY_2^2)\}$	$\{(\mathbf{1},minY_3^1,maxY_3^1),$ $(\mathbf{2},minY_3^2,maxY_3^2)\}$	$\{(\mathbf{2},minY_4^2,maxY_4^2),$ $(\mathbf{3},minY_4^3,maxY_4^3)\}$	$\{(\mathbf{3},minY_5^3,maxY_5^3)\}$
$[V_i]$	$\{(\mathbf{1},4.11,4.13)\}$	$\{(\mathbf{1},4.12,4.12),$ $(\mathbf{2},6.22,6.32)\}$	$\{(\mathbf{1},4.13,4.13),$ $(\mathbf{2},6.23,6.33)\}$	$\{(\mathbf{2},6.14,6.34),$ $(\mathbf{3},5.24,5.24)\}$	$\{(\mathbf{3},5.15,5.35)\}$

Algorithm 3 DCRC Structure for a Given Set of Time Series

Input: A given reference time series $\mathbf{c}$ of n-length;
Input: A given set of m-length time series $S = \{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_T\}$, with each element, $\mathbf{s}_t$, represented by $\mathbf{s}_t = [s_{t1}, s_{t2}, \cdots, s_{tm}]$, where $t = 1, 2, \cdots, T$.
Output: DCRC structure $\mathbb{V}$.

If $\mathbf{c} = \mathbf{nil}$, let $\mathbf{c} = \mathcal{AS}(\mathbf{s}_k, n)$ $(n < m)$ by Algorithm ;
Initialise series $\mathbb{V} = [\{\}, \{\}, \cdots, \{\}]$ of m-length;
for $t = 1$ to T **do**
  Let $\mathbf{r}_t = \mathcal{R}(\mathbf{s}_t, \mathbf{c})$ by Algorithm 1;
  for $i = 1$ to m **do**
    if $(r_{ti} \in \mathbb{V}.P_i)$ **then**
      Let $\mathbf{v} = \mathbb{V}.\mathbf{v}_i^{r_{ti}}$;
      Let $\mathbf{v}.l = min(s_{ti}, \mathbf{v}.l)$;
      Let $\mathbf{v}.u = max(s_{ti}, \mathbf{v}.u)$;
    else
      Let $V_i = V_i \cup \{(r_{ti}, s_{ti}, s_{ti})\}$;
    end if
  end for
end for
return $\mathbb{V}$

4 Time Series Indexing with DCRC

4.1 DCRC based DTW Lower Bound (LB_DCRC)

Given set S of m-length times series and a DCRC structure $\mathbb{V}$ determined by Equ. (10), a lower bound of DTW from a given time series $\mathbf{q}$ to the elements of S can be defined as the minimal DTW distance from $\mathbf{q}$ to the elements of $Cover(\mathbb{V})$, as defined in Equ. (11).

$$LB\_DCRC(\mathbf{q}, S) = \min_{\mathbf{x} \in Cover(\mathbb{V})} DTW(\mathbf{q}, \mathbf{x}) \tag{11}$$

The DCRC based lower bound of classic DTW, namely, LB_DCRC, is summarized in Algorithm 4. Given the ratio of the width of the Sakoe-Chiba Band to the length of the time series, denoted by λ, the time complexity for the algorithm is $O(\lambda m^2 n)$.

Note that, in a given DCRC structure $\mathbb{V}$, the number of feasible relationships grows with the power of m and n, i.e. is $O(\phi^{mn})$, where ϕ is a positive constant. However, the computation of LB_DCRC does not directly enumerate all the relationships, and achieves polynomial complexity by using dynamic programming.

In Algorithm 4, $\sqrt{a_{ijk}}$ represents the lower bound DTW from i-length time series $\mathbf{q}[1:i]$ to j-length DCRC $\mathbb{V}'(V_1', V_2', \cdots, V_j')$, satisfying $V_l' = \{\mathbf{v} \in V_k | \mathbf{v}.p \leq k\}$, for $l = 1, 2, \cdots, j$. Then a_{ijk} is computed by the recursive formula at Line 16.

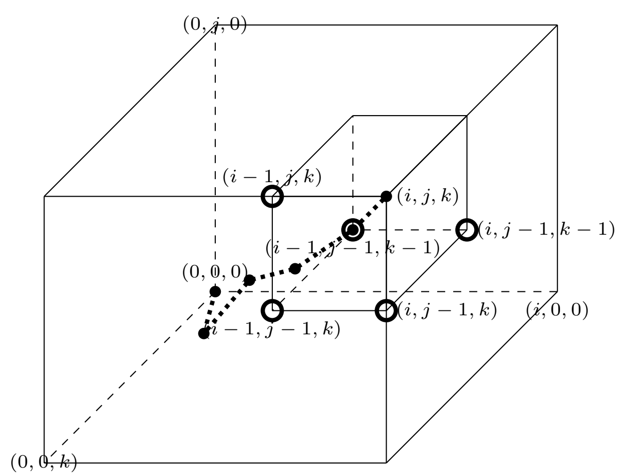


Figure 4: A feasible matching path from (0,0,0) to (i, j, k) for the computation of LB_DCRC

We will prove Algorithm 4 satisfies Equ. (11) by Theorem 4 in Sec. 5. In Fig. 4, the dotted line shows a solution for LB_DCRC. The computation of point (i, j, k) depends on the five points $(i-1, j-1, k)$, $(i-1, j-1, k-1)$, $(i-1, j, k)$, $(i, j-1, k)$ and $(i, j-1, k-1)$. Let (i_1, j_1, k_1), (i_2, j_2, k_2), $\cdots$, (i_L, j_L, k_L) be an optimized path and $\mathbf{g}$ $[g_1, g_2, \cdots, g_m]$ the optimized time series

Algorithm 4 DCRC based lower bound of DTW (LB_DCRC)

Input: Set S of m-length times series;
Input: DCRC structure $\mathbb{V} = [V_1, V_2, \cdots, V_m]$ satisfying $S \subset Cover(\mathbb{V})$;
Input: Ratio λ $(0 < \lambda \leq 1)$ of band width to m; an m-length query time series $\mathbf{q} = [q_1, q_2, \cdots, q_m]$.
Output: $LB\_DCRC(\mathbf{q}, S)$.

Let $\mathbf{A} = [a_{ijk}]$ be an $m \times m \times n$-size array, each $a_{ijk} = +\infty$ initially;
Let B be an empty set;
for $i = 1$ **to** m, $j = 1$ **to** m **do**
    if $|i - j| \leq \lambda m$ **then**
        for each k **in** $\mathbb{V}.P_j$ **do**
            $B = B \cup (i, j, k)$;
        end for
    end if
end for
for each (i, j, k) **in** B **do**
    Let $\eta_1 = \alpha(i-1, j-1, k)$;
    Let $\eta_2 = \alpha(i-1, j-1, k-1)$;
    Let $\eta_3 = \alpha(i-1, j, k)$;
    Let $\eta_4 = \alpha(i, j-1, k)$;
    Let $\eta_5 = \alpha(i, j-1, k-1)$;
    Let $a_{ijk} = min(\eta_1, \eta_2, \eta_3, \eta_4, \eta_5) + \gamma(i, j, k)$;
end for
return $\sqrt{a_{mmn}}$

function $\alpha(i, j, k)$
    if $i = j = k = 0$ **then return** 0;
    else if $(i, j, k) \in B$ **return** a_{ijk};
    else return $+\infty$;
    end if
end function

function $\gamma(i, j, k)$
    Let $x = q_i$;
    Let $y_0 = \mathbb{V}.L_j^k$;
    Let $y_1 = \mathbb{V}.U_j^k$;
    if $x < y_0$ **return** $(y_0 - x)^2$;
    else if $x > y_1$ **return** $(x - y_1)^2$;
    else return 0;
    end if
end function

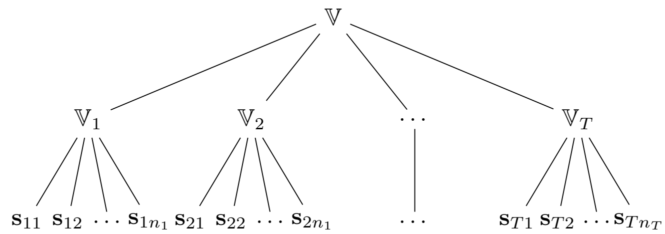


Figure 5: Illustration of Hierarchical DCRC with two layers

in Equ. (11). As $j_p = j_q \Rightarrow k_p = k_q (p \neq q)$, assume $\mathbf{r} = [r_1, r_2, \cdots, r_m]$ satisfying for $\forall (p \in \{1, 2, \cdots, m\}\ \exists q (j_q = p \wedge k_q = r_p)$, we have $g_p \in [L_p^{r_p}, U_p^{r_p}]$ for $p = 1, 2, \cdots, m$. Furthermore, (i_1, j_1), (i_2, j_2), $\cdots, (i_L, j_L)$ is exactly the DTW path between the query time series $\mathbf{q}$ and the optimal solution $\mathbf{g}$.

4.2 Hierarchical DCRC (HDCRC)

Consider a given series of sets $S_1, S_2, \cdots, S_T$, where S_t $(t = 1, 2, \cdots, T)$ is a set of m-length time series; and a given series of DCRC structures $\mathbb{V}_1, \mathbb{V}_2, \cdots, \mathbb{V}_T$, where $\mathbb{V}_t.n = n$ and $S_t \subseteq Cover(\mathbb{V}_t)$ for $t = 1, 2, \cdots, T$.

The problem is how to obtain a DCRC structure $\mathbb{V}$ satisfying $\bigcup_{t=1}^{T} S_t \subseteq Cover(\mathbb{V})$ and $\mathbb{V}.n = n'$ $(n' \leq n)$ according to $\mathbb{V}_1$, $\mathbb{V}_2$, $\cdots$, $\mathbb{V}_T$ only, and not the entire set of elements of $S_1, S_2, \cdots, S_T$. The hierarchical structure is illustrated in Fig. 5. Algorithm 5 sets out the procedure for determining the DCRC structure.

At line 3, the reference time series $\mathbf{c}$ of length n' is converted from the reference $\mathbf{x}_1$ of $\mathbb{V}_1$ by Algorithm 2. The components of $\mathbb{V}$ are built by the steps from Lines 9-19. For the i-th set in $\mathbb{V}$, if $j \in \mathbb{V}_t.P_i$, we have $r_j \in \mathbb{V}.P_i$.

Algorithm 5 Hierarchical DCRC

Input: Time series $\mathbf{c}$ of n'-length
Input: Set $(\mathbb{V}_1, \mathbb{V}_2, \cdots, \mathbb{V}_T)$ of m-length DCRC structures, where $\mathbb{V}_t.n = n(n' \leq n)$ for $t = 1, 2, \cdots, T$.
Output: A DCRC structure $\mathbb{V}$ satisfying $\bigcup_{t=1}^{T} Cover\ (\mathbb{V}_t) \subseteq Cover\ (\mathbb{V})$ and $\mathbb{V}.n = n'$.

```
1:  if c = nil then
2:      Let x_1 be the reference time series of V_t.
3:      Let c = AS(x_1, n') by Algorithm 2;
4:  end if
5:  Initialise V = {V_1, V_2, ..., V_m} such that V_i = φ for i = 1, 2, ..., m;
6:  for t = 1 to T do
7:      Let x_t be the reference time series of V_t.
8:      Let r[r_1, r_2, ..., r_n] = R(x_t, c);
9:      for i = 1 to m do
10:         for each j in V_t.P_i do
11:             Let k = r_j;
12:             if (k ∈ V.P_i) then
13:                 Let V.L_i^k = min(V.L_i^k, V.L_ti^j);
14:                 Let V.U_i^k = max(V.U_i^k, V.U_ti^j);
15:             else
16:                 Let V_i = V_i ∪ {(k, V.L_ti^j, V.U_ti^j)}
17:             end if
18:         end for
19:     end for
20: end for
21: return V
```

4.3 DCRC-Tree and Relevant Functions

Based on the HDCRC structure, an R-tree [8] like indexing tree, named DCRC-tree, is proposed for efficient querying. Each node in a DCRC-tree corresponds to a DCRC structure $\mathbb{V}$ (See Sec. 3.5), rather than a minimal boundary rectangle (MBR) as used in R-trees. When searching a time series from the DCRC-tree, we still adopt the classic DTW (with global constraints).

A tree node of the DCRC-tree is represented by tuple $\mathcal{N}(d, \mathbb{V}, \mathbf{c}, \mathcal{P}arent, \mathcal{C}hildren, \mathcal{S}eries)$, where the components are as defined in Table 2. The relevant basic operators of the DCRC-Tree are given in Table 3.

Table 2: Relevant functions of DCRC-Tree

Components	Description
d	Depth of the tree node
$\mathbb{V}$	DCRC structure
$\mathbf{c}$	Referent time series
$\mathcal{P}arent$	Parent node
$\mathcal{C}hildren$	Child nodes
$\mathcal{S}eries$	Chiled time series

The implementation of function $create\_dcrc(\mathbf{c}, S)$ utilizes Algorithm 3 with $\mathbf{c}, S$ as input parameters. The implementation of function $update\_dcrc(\mathbb{X}, \mathbf{c}, \mathbf{x})$ is derived from lines 5 - 13 in Algorithm 3, with $\mathbb{V}, \mathbf{c}, \mathbf{s}_t$ replaced by parameters $\mathbb{X}, \mathbf{c}, \mathbf{x}$. The implementation of function $update\_hdcrc(\mathbb{X}, \mathbf{c}, \mathbb{Y})$ is derived from lines 9 - 19 in Algorithm 5, with $\mathbb{V}, \mathbf{c}, \mathbb{V}_t$ replaced by parameters $\mathbb{X}$, $\mathbf{c}$ and $\mathbb{Y}$.

The implementation of $insert\_\ series(\mathcal{N}, \mathbf{s})$ is as follows:

(a) If $\mathcal{N}.\mathbf{c} = \mathbf{nil}$, then $\mathcal{N}.\mathbf{c}$ is assigned to $\mathcal{AS}(\mathbf{s}, |\mathcal{N}.\mathbf{c}|)$;

(b) Let $\mathcal{N}.\mathcal{S}eries = \mathcal{N}.\mathcal{S}eries \cup \{\mathbf{s}\}$;

(c) Let $\mathcal{N}.\mathbb{V} = update\_dcrc(\mathcal{N}.\mathbb{V}, \mathcal{N}.\mathbf{c}, \mathbf{s})$.

The implementation of $insert\_node(\ \mathcal{N}, \mathcal{N}')$ is as follows:

(a) If $\mathcal{N}.\mathbf{c} = \mathbf{nil}$, then let $\mathcal{N}.\mathbf{c} = \mathcal{AS}(\mathcal{N}'.\mathbf{c}, |\mathcal{N}.\mathbf{c}|)$;

(b) Let $\mathcal{N}.\mathcal{C}hildren = \mathcal{N}.\mathcal{C}hildren \cup \{\mathcal{N}'\}$;

(c) Let $\mathcal{N}.\mathbb{V} = update\_hdcrc(\mathcal{N}.\mathbb{V}, \mathcal{N}.\mathbf{c}, \mathcal{N}'.\mathbb{V})$;

(d) Let $\mathcal{N}'.\mathcal{P}arent = \mathcal{N}$.

4.4 Node Splitting and Insertion in a DCRC-Tree

Motivated by the idea of node splitting in R-trees, we develop a node splitting algorithm for DCRC-trees. Let M be the maximal number of child nodes (not including leaves) of each tree node. There are two cases of node splitting.

The first case is when node $\mathcal{N}$ is a leaf node satisfying $|\mathcal{N}.\mathcal{S}eries| = M$, then it is split into nodes $\mathcal{N}_1$ and $\mathcal{N}_2$, with both $\mathcal{N}_1.\mathcal{S}eries$ or $\mathcal{N}_2.\mathcal{S}eries$ containing $M/2$ time series. Algorithm 6 details the node splitting algorithm.

The second case is is when $\mathcal{N}$ is a none-leaf node satisfying $|\mathcal{N}.\mathcal{C}hildren| = M$, then it is split into nodes $\mathcal{N}_1$ and $\mathcal{N}_2$, such that $\mathcal{N}_1.\mathcal{C}hildren$ and $\mathcal{N}_2.\mathcal{C}hildren$ respectively contain $M/2$ tree nodes. The corresponding node splitting algorithm for the set of tree nodes is similar to Algorithm 6.

Algorithm 7 summarizes the steps for inserting a time series into a given DCRC-tree. These are similar to the steps used with R-trees. From the root, the child node with the minimal increasing volume is selected recursively, until the current node is a leaf. Then, the time series is inserted into the leaf node, and from bottom to top, the parent node is split if the number of its children exceeds a pre-given maximal limit, and the depth of the tree is less than a pre-given maximal limit. Therefore the leaf nodes might have a huge number of time series, which are relatively similar to each other in terms of DTW distance.

For R-Tree and DCRC-Tree, consider the tree node covering a set of time series. In a tree node of a R-Tree:

(1) The covering set is a MBR, each i-th component is a range interval derived from the bands with the i-th entry centered.

(2) The volume is the production of each i-th range interval. When the elements are similar, but have large time axis deformation, we have relatively large volume.

(3) The lower bound DTW to a given query time series, is computed by different Hausdorff-distance-like methods, including LB_Keogh [14], LB_NEW [29], LB_ENHANCED [32], etc.

In a tree node of a DCRC-Tree:

(1) The covering set is a DCRC structure, each i-th component is a set of tuples, and each tuple is a range interval and a subscript.

(2) The volume is computed a defined in Equ. (9). When the elements are similar, but have large time axis deformation, as long as the reference time series is similar to these elements, we have relatively small volume.

(3) The lower bound DTW to a given query time series, is computed by LB_DCRC using a dynamic programming method.

Hence, the DCRC-Tree based on HDCRC is a tighter structure for covering time series samples, than an R-Tree like structure. Consequently, this leads to more efficient pruning when performing a query.

Algorithm 6 Node Splitting for Time Series Set

Input: DCRC-Tree node $\mathcal{N}(|\mathcal{N}.\mathcal{S}eries| = M, \mathcal{N}.d < d_{max})$.
Output: The updated DCRC-Tree nodes $\mathcal{N}$ and $\mathcal{N}'$ after splitting.

Let $v_{ol} = volume(\mathcal{N})$;
Let $\mathbb{X}_i = create\_dcrc(\mathcal{N}.\mathbf{c}, \{\mathcal{N}.\mathcal{S}eries[i]\})$, for $i = 1, 2, \cdots, M$;
Let $j_1 = \operatorname*{argmin}_i volume(\mathbb{X}_i)$, $j_2 = \operatorname*{argmax}_i volume\ (\mathbb{X}_i)$;
Let $\mathbf{x}_1 = \mathcal{N}.\mathcal{S}eries[j_1]$, and let $\mathbf{x}_2 = \mathcal{N}.\mathcal{S}eries[j_2]$;
Let $\mathcal{N}.\mathcal{S}eries = \phi$;
Create a new tree node $\mathcal{N}'$, let $|\mathcal{N}'.\mathbf{c}| = |\mathcal{N}.\mathbf{c}|$, and $|\mathcal{N}'.d = |\mathcal{N}.d$;
$insert\_series(\ \mathcal{N}, \mathbf{x}_1)$;
if $v_{ol} < \varepsilon$ **then**
  Let $\mathcal{N}'.\mathbf{c} = \mathcal{N}.\mathbf{c}$;
end if
$insert\_series(\ \mathcal{N}', \mathbf{x}_2)$;
Let $S' = \mathcal{N}.\mathcal{S}eries - \{\mathbf{x}_1\} - \{\mathbf{x}_2\}$;
for each s **in** S' **do**
  Let $v_1(\mathbf{s}) = volume(create\_dcrc(\mathbf{x}_1, \{\mathbf{x}_1, \mathbf{s}\}))$;
  Let $v_2(\mathbf{s}) = volume(create\_dcrc(\mathbf{x}_2, \{\mathbf{x}_2, \mathbf{s}\}))$;
  Denote $\omega(\mathbf{s}) = v_1(\mathbf{s}) - v_2(\mathbf{s})$;
end for
Let $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{M-2}$ be the permutation of the elements of S' satisfying $|\omega(\mathbf{y}_i)| \geq |\omega(\mathbf{y}_{i+1})|$ for $i = 1, 2, \cdots, M-3$;
for $i = 1$ **to** $M - 2$ **do**
  if $|\mathcal{N}.\mathcal{S}eries| = M/2$ **then**
    $insert\_series(\mathcal{N}', \mathbf{y}_i)$;
  else if $|\mathcal{N}'.\mathcal{S}eries| = M/2$ **then**
    $insert\_series(\mathcal{N}, \mathbf{y}_i)$;
  else
    if $\omega(\mathbf{s}) < 0$ **then**
      $insert\_series(\mathcal{N}, \mathbf{y}_i)$;
    else
      $insert\_series(\mathcal{N}', \mathbf{y}_i)$;
    end if
  end if
end for
return $\mathcal{N}, \mathcal{N}'$

Algorithm 7 Insertion in a DCRC-tree

Input: Time series s of m-length
Input: Root $\mathcal{T}$ of DCRC-tree.
Output: The updated root $\mathcal{T}$ after insertion.

if $\mathcal{T} = \textbf{nil}$ **then**
New a DCRC-tree node $\mathcal{T}$;
$insert\_series(\mathcal{T}, \mathbf{s})$;
return $\mathcal{T}$;
end if
Let $\mathcal{N} = \mathcal{T}$;
while $\mathcal{N}.\mathcal{C}hildren \neq \phi$ **do**
for each $\mathcal{N}_i$ in $\mathcal{N}.\mathcal{C}hildren$ **do**
Copy $\mathcal{N}_i.\mathbb{V}$ to $\mathbb{X}_i$;
Let $\mathbb{Y}_i = update\_dcrc(\mathbb{X}_i, \mathcal{N}_i.\mathbf{c}, \{\mathbf{s}\})$;
end for
Let $\mathcal{N} = \mathcal{N}.\mathcal{C}hildren[k]$, $k = \underset{i}{\operatorname{argmin}}\, volume(\mathbb{Y}_i)$;
end while
$insert\_series(\mathcal{N}, \mathbf{s})$;
Let $\mathcal{N}' = \textbf{nil}$;
if $\mathcal{T}.d < d_{max}$ and $|\mathcal{N}.\mathcal{S}eries| = M$ **then**
Split node $\mathcal{N}$ into $\mathcal{N}$ and $\mathcal{N}'$;
end if
while true do
Let $\mathcal{N}_t = \mathcal{N}.\mathcal{P}arent$;
if $\mathcal{N}_t = \textbf{nil}$ **then**
if $\mathcal{N}' \neq \textbf{nil}$ **then**
Create a new node $\mathcal{T}$, let $\mathcal{T}.\mathcal{P}arent = \textbf{nil}$;
Let $\mathcal{T}.d = \mathcal{N}.d + 1$
Let $|\mathcal{T}.\mathbf{c}| = |\mathcal{N}.\mathbf{c}|/2$;
$insert\_node(\mathcal{T}, \mathcal{N})$;
$insert\_node(\mathcal{T}, \mathcal{N}_b)$;
end if
return $\mathcal{T}$;
else
if $\mathcal{N}' = \textbf{nil}$ **then**
Update $\mathcal{N}_t.\mathbb{V}$ with $\mathcal{N}_t.\mathcal{C}hildren$ by Algorithm 5;
else
$insert\_node(\mathcal{N}_t, \mathcal{N}')$;
if $|\mathcal{N}_t.\mathcal{C}hildren| = M$ **then**
Split node $\mathcal{N}$ into $\mathcal{N}$ and $\mathcal{N}'$;
end if
end if
Let $\mathcal{N} = \mathcal{N}_t$;
end if
end while

Table 3: Relevant functions of DCRC-Tree

Function	Input/Output	Description
$create\_dcrc$	$\mathbf{c}$	Reference time series
	S	Set of time series
	-	A new DCRC structure built from S
$update\_dcrc$	$\mathbb{V}$	Original DCRC structure
	$\mathbf{c}$	Reference time series
	$\mathbf{x}$	Newly inserted Time series
	-	The updated DCRC structure $\mathbb{V}$ after insertion of $\mathbf{x}$
$update\_hdcrc$	$\mathbb{X}$	Original DCRC structure
	$\mathbf{c}$	Reference time series
	$\mathbb{Y}$	Newly inserted DCRC
	-	The updated DCRC structure $\mathbf{X}$ after insertion of $\mathbb{Y}$
$insert\_series$	$\mathcal{N}$	Original DCRC-Tree Node
	$\mathbf{s}$	Newly inserted Time series
	-	The updated DCRC-Tree node $\mathcal{N}$ after insertion of s
$insert\_node$	$\mathcal{N}$	Original DCRC-Tree Node
	$\mathcal{N}'$	Newly inserted Node
	-	The updated DCRC-Tree node $\mathcal{N}$ after insertion of $\mathcal{N}'$

5 Theorems for DCRC

For the algorithms in Secs. 3 and 4.2, we will prove their correctness and efficiency in this section. Theorem 1 assures the DCRC structure can cover a given set. Theorems 2 and 3 prove the tightness of the DCRC covering. Considering the lower bound of DTW between a given query time series and a given DCRC structure by Algorithm 4, Theorem 4 proves its correctness and Theorem 5 proves that the hierarchical structure generated by Algorithm 5 is still a DCRC structure, which is used to generate an indexing tree.

Theorem 1 *For a given set S of m-length time series, let $\mathbb{V}$ be the return value of Algorithm 3, then $S \subseteq Cover(\mathbb{V})$.*

Proof 1 *Given $\mathbf{s}_t = [s_{t1}, s_{t2}, \cdots, s_{tm}] \in S$ where $t \in \{1, 2, \cdots, T\}$, let $\mathbf{r}[r_1, r_2, \cdots, r_m]$ be the ACM-relationship at line 4 in Algorithm 3. From the loop from lines 5 to 13, we have $s_{ti} \in [L_i^{r_i}, U_i^{r_i}]$ for $i = 1, 2, \cdots, m$. From $\mathbf{r} \in \mathbb{R}(m, n)$ (defined in Definition 1), and the definition of $Rect_r(\mathbb{V}, \mathbf{r})$, $\mathbf{s}_t \in Rect_r(\mathbb{V}, \mathbf{r})$, i.e., $\mathbf{s}_t \in Cover(\mathbb{V})$ from Equ.* (10).

Lemma 1 *Let $\mathbf{x}_1 = [x_{11}, x_{12}, \cdots, x_{1m_1}]$, $\mathbf{x}_2 = [x_{21}, x_{22}, \cdots, x_{2m_2}]$ be two given time series, of length $m_1, m_2 \quad (m_1 \leq m_2)$, and let y be a constant. If we have $\alpha = \sqrt{\sum_{i=1}^{m_1}(x_{1i} - y)}$ and $\beta = \sqrt{\sum_{i=1}^{m_2}(x_{2i} - y)}$, we have $DTW^2(\mathbf{x}_1, \mathbf{x}_2) \leq 2\lceil m_2/m_1 \rceil(\alpha^2 + \beta^2)$.*

Proof 2 *Denote $d = DTW(\mathbf{x}_1, \mathbf{x}_2)$. Consider a matching path $\mathbf{W}$ of length m_2 (might not be a DTW warping path) from $\mathbf{x}_2$ to $\mathbf{x}_1$, such as $(1, i_1)$, $(2, i_2)$, $\cdots$, (m_2, i_{m_2}), where $i_k = \lceil km_1/m_2 \rceil$. We have $d^2 \leq \sum_{k=1}^{m_2} (x_{1i_k} - x_{2k})^2 \leq \sum_{k=1}^{m_2} 2((x_{1i_k} - y)^2 + (x_{2k} - y)^2)$. As $i_k = \lceil km_1/m_2 \rceil$, we have $d^2 \leq 2\sum_{k=1}^{m_2}(x_{2k} - y)^2 + 2\lceil m_2/m_1 \rceil \sum_{k=1}^{m_1}(x_{1k} - y)^2 \leq 2\lceil m_2/m_1 \rceil(\alpha^2 + \beta^2)$. Therefore, $DTW^2(\mathbf{x}_1, \mathbf{x}_2) \leq 2\lceil m_2/m_1 \rceil(\alpha^2 + \beta^2)$.*

Consider three time series $\mathbf{x}_1 = [x_{11}, x_{12}, \cdots, x_{1m_1}]$, $\mathbf{x}_2 = [x_{21}, x_{22}, \cdots, x_{2m_2}]$ and $\mathbf{y} = [y_1, y_2, \cdots, y_n]$ of length m_1, m_2 and n, respectively, with $n < m_1, m_2$.

Theorem 2 *If $\mathcal{D}(\mathbf{x}_1, \mathbf{y}) = \alpha$, $\mathcal{D}(\mathbf{x}_2, \mathbf{y}) = \beta$ (where function $\mathcal{D}$ is defined in Equ.* (3)*), we have $DTW(\mathbf{x}_1, \mathbf{x}_2) \leq \sqrt{2(m_2 - n)(\alpha^2 + \beta^2)}$.*

Proof 3 *Let $\mathbf{r}_1[r_{11}, r_{12}, \cdots, r_{1m}] = \mathcal{R}(\mathbf{x}_1, \mathbf{y})$, let $\mathbf{r}_2[r_{21}, r_{22}, \cdots, r_{2m}] = \mathcal{R}(\mathbf{x}_2, \mathbf{y})$, and let $\mathbf{W}$ denote a matching path from $\mathbf{x}_1$ to $\mathbf{x}_2$, which is divided into n segments. Let the t-th segment correspond to set $X_{pt} = \{k | \ r_{pk} = t\}$, and let a_{pt}, b_{pt} denote the minimum and maximum of X_{pt}, respectively, where $p = 1, 2$.*

Let $\alpha_t^2 = \sum_{k=a_{1t}}^{b_{1t}}(x_{1k} - y_t)^2$ and $\beta_t^2 = \sum_{k=a_{2t}}^{b_{2t}}(x_{2k} - y_t)^2$.

We have $1 \leq |X_{1t}|, |X_{2t}| \leq m_2 - n$. From Lemma 1, we have $DTW^2(\mathbf{x}_1(a_{1t} : b_{1t}), \mathbf{x}_2(a_{2t} : b_{2t})) \leq 2\lceil d_1/d_0 \rceil \ (\alpha_t^2 + \beta_t^2)$. Then $DTW^2(\mathbf{x}_1, \mathbf{x}_2) \leq \sum_{t=1}^{n} DTW^2(\mathbf{x}_1 \ (a_{1t} : b_{1t}), \mathbf{x}_2(a_{2t} : b_{2t})) \leq 2(m_2 - n) \sum_{t=1}^{n} (\alpha_t^2 + \beta_t^2) = 2(m_2 - n)(\alpha^2 + \beta^2)$. Then $DTW(\mathbf{x}_1, \mathbf{x}_2) \leq \sqrt{2(m_2 - n)(\alpha^2 + \beta^2)}$.

Theorem 3 *Considering Algorithm 3, let set $S = \{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_T\}$ of m-length time series and time series $\mathbf{c}$ of n-length be the input parameters, and let $\mathbb{V}$ be the output DCRC structure. Assume $\mathcal{D}(\mathbf{s}_t, \mathbf{c}) \leq \alpha$ for $\forall t \in \{1, 2, \cdots, T\}$. If $\mathbf{x} = [x_1, x_2, \cdots, x_m] \in Cover(\mathbb{V})$, then $\mathcal{D}(\mathbf{x}, \mathbf{c}) \leq \sqrt{m}\alpha$. $\mathcal{D}$ is defined in Equ.* (3).

Proof 4 *Firstly, we will prove that for $\forall \mathbb{V}.\mathbf{v}_i^j$ (assumed $\mathbf{v}$), we have $\mathbf{v}.l \geq c_j - \alpha$ and $\mathbf{v}.u \leq c_j + \alpha$, where c_j is the j-th entry of $\mathbf{c}$. From the computation of warping path $\mathbf{W}$ at line 4, and the assumption $\mathcal{D}^2(\mathbf{s}_t, \mathbf{c}) = \sum_{i=1}^{m}(s_{ti} - c_{r_i})^2 \leq \alpha^2$, we have $c_{r_i} - \alpha \leq s_{ti} \leq c_{r_i} + \alpha$. For each $\mathbf{v}(\mathbb{V}.\mathbf{v}_i^j)$, from the assignments at lines 7-9 and 11, we have $\mathbf{v}.l \geq c_j - \alpha$ and $\mathbf{v}.u \leq c_j + \alpha$.*

If $\mathbf{x} \in Cover(\mathbb{V})$, there exists a series $\mathbf{r} = [r_1, r_2, \cdots, r_m] \in \mathbb{R}(m, n)$ satisfying $\mathbf{x} \in Rect_r(\mathbb{V}, \mathbf{c})$. From the definition of $\mathcal{D}$, $\mathcal{D}^2(\mathbf{x}, \mathbf{c}) \leq \sum_{i=1}^{m}(x_i - c_{r_i})^2$. From Equ. (8)*, we have $x_i \in [\mathbf{v}.l, \mathbf{v}.u]$. As $\mathbf{v}.l \geq c_{r_i} - \alpha$ and $\mathbf{v}.u \leq c_{r_i} + \alpha$, then $\mathcal{D}^2(\mathbf{x}, \mathbf{c}) \leq \sum_{i=1}^{m}(x_i - c_{r_i})^2 \leq m\alpha^2$. Then $\mathcal{D}(\mathbf{x}, \mathbf{c}) \leq \sqrt{m}\alpha$.*

From Theorems 2, 3, and Algorithm 3, we can conclude that if the elements of DCRC are all similar to the reference $\mathbf{c}$ as measured by function $\mathcal{D}$, the elements are also similar to each other in terms of the DTW Distance.

Theorem 4 *The return value of Algorithm 4 is $LB\_DCRC(\mathbf{q}, S)$ as defined in Equ.* (11).

Proof 5 *Firstly, we will prove $a_{ijk} = \min_{\mathbf{x}} DTW(\mathbf{q}(1:i), \mathbf{x}(1:j))$ s.t. $\mathbf{x}(1:j) \in Cover(\mathbb{V}_j)$ and $x_j \in [L_j^k, U_j^k]$, where $\mathbb{V}_j = [V_1, V_2, \cdots, V_j]$ and $\mathbf{x}(1:j) = [x_1, x_2, \cdots, x_j]$.*

Mathematical induction. Assume $a_{i'j'k'}$ satisfy the above min equation for $\forall(i', j', k')$ $((i' \leq i \wedge j' \leq j \wedge k' \leq k)$ $((i', j', k') \neq (i, j, k)))$. We will prove a_{ijk} also satisfies the above min equation.

In line 16, a_{ijk} is recursively represented by the sum of $\gamma(i, j, k)$ and $a_{i'j'k'}$. Considering subscript pair $(i'j')$ of $a_{i'j'k'}$, there are three cases: $(i-1, j-1)$, $(i-1, j)$ and $(i, j-1)$. In the case of $(i', j') = (i-1, j-1)$, from the definition of $Cover$ in Equ. (10) *and the ACM-relationships in Definition 1, we have $(k-1) \in \mathbb{V}.P_{j-1}$ or $k \in \mathbb{V}.P_{j-1}$. The two cases correspond to η_1 and η_2, respectively. Similarly, η_2, η_3, η_4 and η_5 correspond to the other cases.*

Note that $\eta_6 = \alpha(i-1, j, k-1)$ and $\eta_7 = \alpha(i, j, k-1)$ are excluded. Considering a_{ijk} is the lower bound of DTW from $\mathbf{q}(1:i)$ to $\mathbf{x}(1:j)$. As the optimum $\mathbf{x} \in Cover(\mathbb{V})$, there exists $\mathbf{r} = [r_1, r_2, \cdots, r_j] \in \mathbb{R}(j, k)$ satisfying that $x_t \in \mathbb{V}.\mathbf{v}_t^{r_t}$ for $t = 1, 2, \cdots, j$. If η_6 or η_7 is adopted in the computation of a_{ijk}, then $(j, k-1)$ and (j, k) will appear in $r_1, r_2, \cdots, r_j$ at the same time, which contradicts the definition of ACM-Relationship.

Using dynamic programming, a_{ijk} also satisfies the minimal assumption. Finally, $\sqrt{a_{mmn}}$ at line 18 is the minimum of Equ. (11).

Theorem 5 *The return value $\mathbb{V}$ of Algorithm 5 satisfies $\bigcup_{t=1}^{T} Cover(\mathbb{V}_t) \subseteq Cover(\mathbb{V})$.*

Proof 6 *For any given $\mathbf{s} = [s_1, s_2, \cdots, s_m] \in Cover(\mathbb{V}_t)$, there exists $\mathbf{b} = [b_1, b_2, \cdots, b_m]$ satisfying $\mathbf{s} \in Rect_r(\mathbb{V}_t, \mathbf{b})$, i.e., $s_i \in \mathbb{V}_t.[L_i^{b_i}, U_i^{b_i}]$ for $i = 1, 2, \cdots, m$. In addition, $\mathbf{b}$ satisfies the ACM-relationships.*

Consider line 8 in Algorithm 5, let $\mathbf{r} = [r_1, r_2, \cdots, r_n]$. From Definition 1, we have that $\mathbf{r}$ satisfies the ACM-relationships.

The series $[r_{b_1}, r_{b_2}, \cdots, r_{b_m}]$ can be shown to satisfy the ACM-relationships as follows. As $r_1 = 1, b_1 = 1, r_n = n'$ and $b_m = n$, then $r_{b_1} = 1$ and $r_{b_m} = n'$, i.e., "Alignment" is satisfied. As $0 \leq r_{i+1} - r_i \leq 1$ for $i = 1, 2, \cdots, n-1$ and $0 \leq b_{i+1} - b_i \leq 1$, for $i = 1, 2, \cdots, m-1$, then $0 \leq r_{b_{i+1}} - r_{b_i} \leq 1$, i.e., "Continuity" and "Monotonicity" are satisfied. From the assignment at lines 12-17, we have $s_i \in \mathbb{V}.[L_i^{r_{b_i}}, U_i^{r_{b_i}}]$, i.e., $\mathbf{s} \in Cover(\mathbb{V})$.

According to Theorem 5, if the LB_DCRC of an upper layer is larger than a given acceptable range query tolerance, then all of its sub-layers can be pruned to reduce computational load.

6 Experiments

In order to illustrate the effectiveness of our algorithms and indexing structure, experiments are carried out in this section. We use LB_NEW [29] and LB_ENHANCED [32] for comparisons. The experiments are divided into two parts, the first part, presented in Sec. 6.2, provides a comparison of the different DTW lower bounds. In addition, we also perform experiments to analyze the impact of parameters including the length of time series, the ratio of the width of the Sakoe-Chiba Band to the length of the time series λ, and acceptable query tolerance ε. The second part, presented in Sec. 6.3, shows the performance of the different index trees.

6.1 Setup

The datasets selected for our experiments are from the UCR Time Series Classification Archive [3]. Firstly, we compute the average of the LB_DCRC distances from the query time series to the DCRC structure using Algorithm 4. Then we compute the average DTW from the query time series to all the samples in the dataset S.

The computed LB_DCRC and actual DTW values for different λ are shown in Table 4. The average lower bound distance of LB_DCRC is lower than DTW for the 20 datasets. The time series have different length m_i. The dimension of $\mathbb{V}$ of the DCRC is set to m_i and the dimension of reference $\mathbf{r}$ of the DCRC is set to $m_i/2$.

Table 4: Average LB_DCRC / DTW values for different λ

Dataset	Dimension	λ=0.2	λ=0.6	λ=1.0
synthetic_control	60	2.189/5.757	1.987/5.603	1.987/5.603
Gun_Point	150	0.317/0.845	0.287/0.820	0.287/0.820
CBF	128	2.497/4.715	2.413/4.645	2.413/4.645
FaceAll	131	2.196/5.743	1.917/5.616	1.906/5.616
OSULeaf	427	1.416/5.543	1.326/5.411	1.326/5.411
SwedishLeaf	128	0.322/1.306	0.319/1.305	0.319/1.305
50Words	270	7.139/8.897	5.002/6.793	4.866/6.686
Trace	275	10.281/10.864	10.051/10.656	10.051/10.656
MedicalImages	99	1.696/3.545	1.379/3.209	1.363/3.204
ShapeletSim	500	8.148/13.396	8.135/13.396	8.135/13.396
FaceFour	350	4.951/7.250	4.794/7.237	4.794/7.237
Lighting2	637	4.858/8.804	3.828/7.842	3.828/7.842
Lighting7	319	6.770/9.794	5.223/8.268	5.195/8.268
FacesUCR	131	3.696/6.407	3.522/6.361	3.494/6.355
Adiac	176	0.899/1.179	0.899/1.179	0.899/1.179
MoteStrain	84	1.560/4.222	1.478/4.048	1.478/4.048
Fish	463	0.416/0.992	0.416/0.992	0.416/0.992
Plane	144	2.762/3.543	2.698/3.485	2.698/3.485
Car	577	0.663/1.243	0.663/1.243	0.663/1.243
Beef	470	3.112/3.908	3.099/3.894	3.099/3.894

6.2 Distance and Tightness

In terms of distance, we compute the average distance between the query time series, and the candidate set of time series using four methods: DTW, LB_NEW, LB_ENHANCED and LB_DCRC. Table 5, which shows the results of the average distance when $\lambda = 0.2$, demonstrates that LB_DCRC achieves better performance than LB_NEW and LB_ENHANCED for all datasets.

Definition 3 (Tightness of the DTW Lower Bound) *Given a method LB of obtaining a lower bound of DTW, a set S of time series, and a query time series $\mathbf{q}$, let the tightness of LB for $\mathbf{q}$ and S be defined as $\frac{LB(\mathbf{q},S)}{\underset{\mathbf{s}\in S}{min} DTW(\mathbf{s},\mathbf{q})}$.*

Using this definition, Fig. 6 shows the average tightness of LB_NEW, LB_ENHANCED, and LB_DCRC for different λ (i.e., 0.2, 0.4, 0.6). From the charts, it is clear that LB_DCRC is superior to LB_ENHANCED and LB_NEW on all datasets. When λ increases, the tightness of LB_NEW and LB_ENHANCED decrease significantly. In contrast, the width of the Sakoe-Chiba band has little impact on LB_DCRC, i.e. when time series has relatively large deformation, LB_DCRC is still a tight lower bound of DTW. The dimensions of these datasets are distributed in the range 60 to 637, but this variation in dimension does not impact the performance of LB_DCRC relative to the other methods.

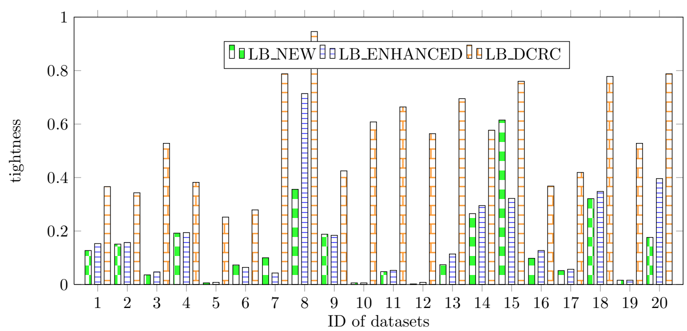
Definition 4 (Pruning Power for a Query Set) *Given a candidate data set S of time series, and a query set of time series Q, the pruning power of LB for set Q is defined as $\frac{|\{\mathbf{q}\in Q|LB(\mathbf{q},S)>\varepsilon\}|}{|Q|}$, where ε is a predefined tolerance.*

Given a tolerance ε, higher pruning power means more query time series can be directly excluded after the computation of the DTW lower bound. Fig. 7 shows a comparison of the pruning power of each approach, with increasing ε. The pruning power of LB_NEW and LB_ENHANCED decrease dramatically, while the decline in LB_DCRC is much more gradual. Fig. 8 shows the average pruning power as a function of ε and the average tightness as a function of λ computed over the datasets.

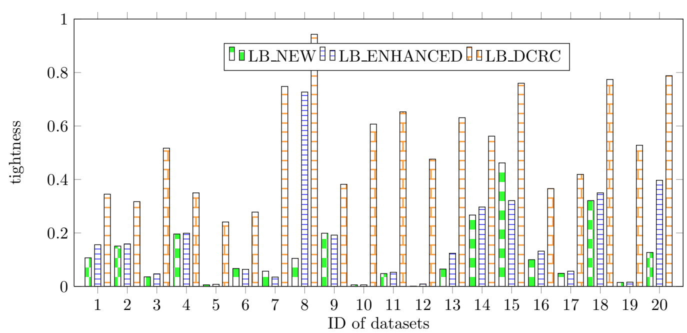
Fig. 9 shows how the tightness changes with the ratio of the Sakoe-Chiba Band for the first 4 datasets employed in our experiments, while Fig. 10 shows the corresponding variation in pruning power as a function of query tolerance. In all cases the curves in Figs. 9 and 10 decrease monotonically and LB_DCRC substantially outperforms its counterparts.

6.3 Indexing Tree Comparisons

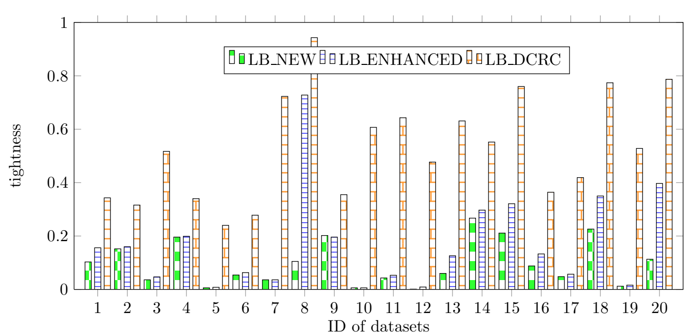
By default, the length of each leave is reduced to 20 by PAA [14]. Let the maximum number of child nodes $M = 20$ and let the maximal depth of the tree $d_{max} = 3$ in Algorithm 7. For each R-Tree node, the maximum number of child nodes M is set to 20. The time series for our experiments are randomly selected from the UCR Archive by the random walk method until the resulting dataset has 1 Gillion bytes. All experiments were optimised and implemented in Ansi C++ and conducted on a 64-bit Win10 operating system with 2.4GHz main frequency, 8 CPUs, 64GB RAM and 4T hard disk.



(a) $\lambda = 0.2$

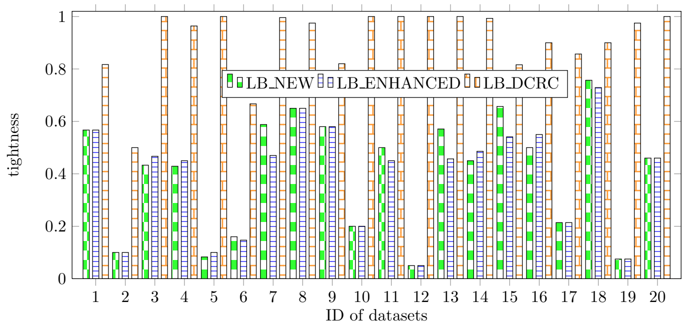


(b) $\lambda = 0.4$

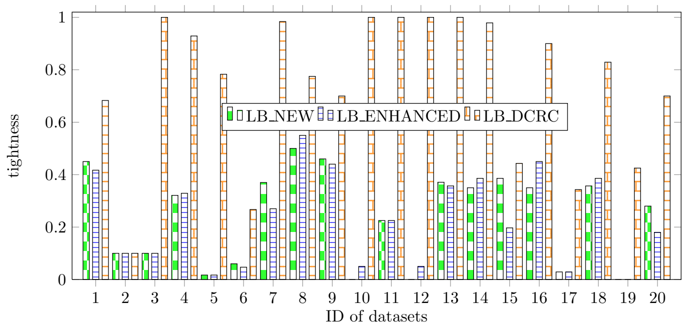


(c) $\lambda = 0.6$

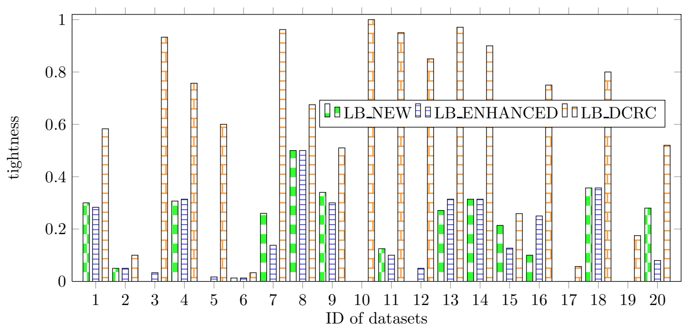
Figure 6: Comparison of lower bound tightness under different ratios λ of warping windows over the 20 datasets



(a) $\varepsilon = 0.1, \lambda = 0.2$

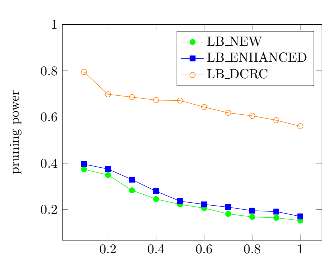


(b) $\varepsilon = 0.5, \lambda = 0.2$

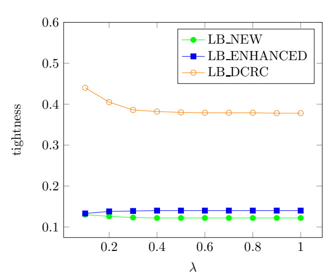


(c) $\varepsilon = 1.0, \lambda = 0.2$

Figure 7: Comparison of pruning power under different acceptable tolerances over the 20 datasets



(a) Pruning power as a function of ε ($\lambda = 1.0$)

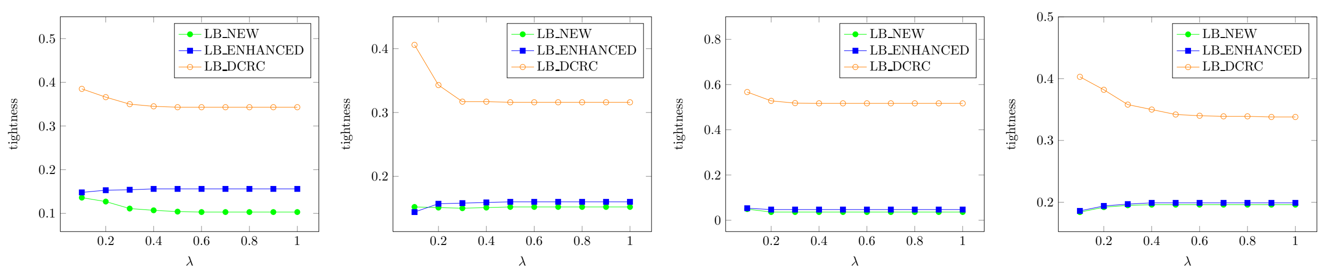


(b) Tightness as a function λ

Figure 8: Pruning power as a function of ε and tightness as a function of λ averaged over the 20 datasets

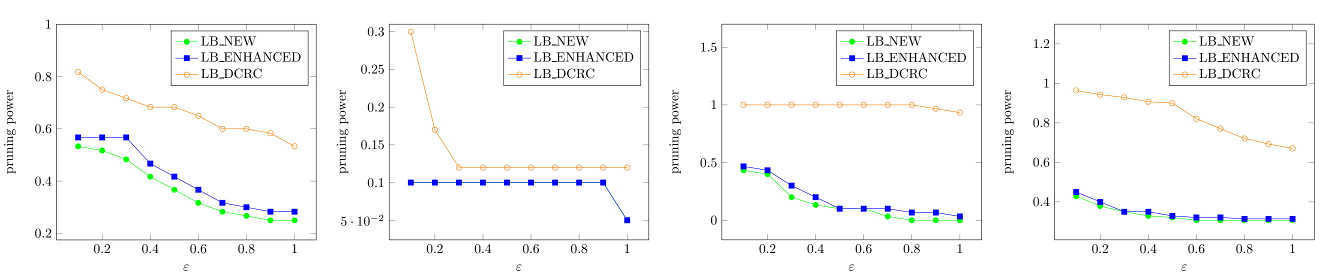
Table 5: Average distance for $\lambda = 0.2$

Dataset	Dimension	DTW	LB_DCRC	LB_NEW	LB_ENHANCED
synthetic_control	60	5.757	2.189	0.771	0.942
Gun_Point	150	0.845	0.317	0.148	0.154
CBF	128	4.715	2.497	0.168	0.218
FaceAll	131	5.743	2.196	1.112	1.127
OSULeaf	427	5.543	1.416	0.033	0.042
SwedishLeaf	128	1.306	0.322	0.079	0.070
50Words	270	8.897	7.139	0.957	0.406
Trace	275	10.864	10.281	3.951	7.755
MedicalImages	99	3.545	1.696	0.807	0.769
ShapeletSim	500	13.396	8.148	0.075	0.082
FaceFour	350	7.250	4.951	0.340	0.368
Lighting2	637	8.804	4.858	0.014	0.070
Lighting7	319	9.794	6.770	0.749	1.068
FacesUCR	131	6.407	3.696	1.689	1.885
Adiac	176	1.179	0.899	0.710	0.386
MoteStrain	84	4.222	1.560	0.416	0.544
Fish	463	0.992	0.416	0.056	0.060
Plane	144	3.543	2.762	1.088	1.197
Car	577	1.243	0.663	0.020	0.021
Beef	470	3.908	3.112	0.659	1.346



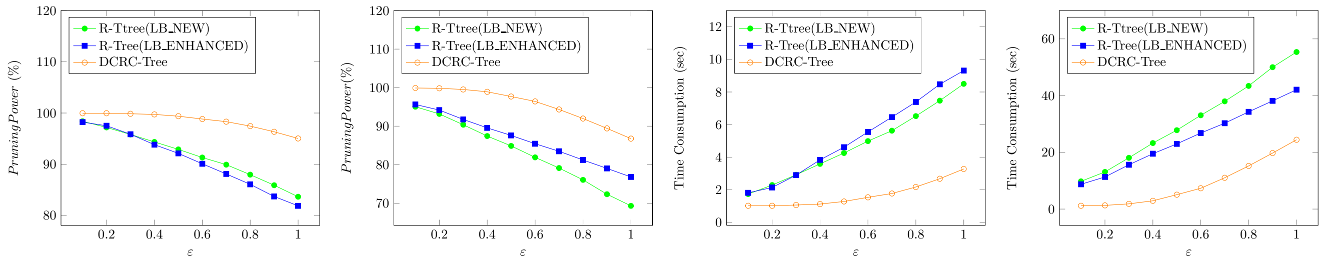
(a) Synthetic Control (b) Gun Point (c) CBF (d) Face All

Figure 9: The relationship between tightness and warping window for 4 selected datasets



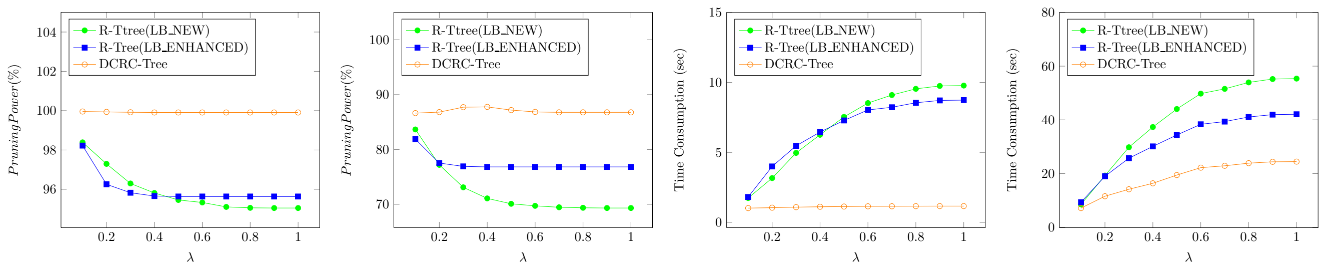
(a) Synthetic Control (b) Gun Point (c) CBF (d) Face All

Figure 10: The relationship between pruning power and the query tolerance for 4 selected datasets ($\lambda = 1.0$)



(a) Pruning power for varying of query tolerance ε (λ=0.1) (b) Pruning power for varying of query tolerance ε (λ=1.0) (c) Computation Time for varying of query tolerance ε (λ=0.1) (d) Computation Time for varying of query tolerance ε (λ=1.0)

Figure 11: The impact of query tolerance on indexing performance for $\lambda = 0.1$ and $\lambda = 1.0$



(a) Pruning power for varying of warping window λ (ε = 0.1) (b) Pruning power for varying of warping window λ (ε = 1.0) (c) Computation Time for varying of warping window λ (ε = 0.1) (d) Time Consumption for varying of warping window λ (ε = 1.0)

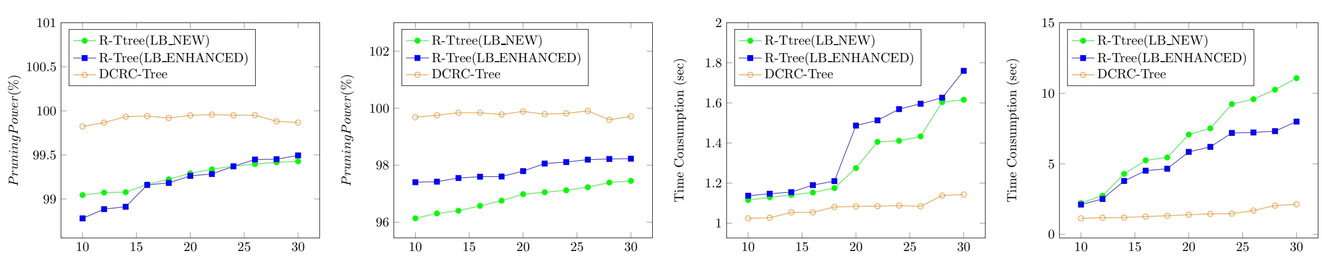
Figure 12: Indexing performance comparisons with different warping windows when $\varepsilon = 0.1$ and $\varepsilon = 1.0$

For LB_NEW and LB_ENHANCED, we construct the corresponding index structures as R-trees, while for LB_DCRC we use a DCRC-tree. If the depth of the DCRC-tree in Algorithm 7 reaches the given maximum, time series contained in the leaf nodes will not be split, i.e., a leaf node might contain a huge number of time series which need to be stored in the same hard disk file.

Fig. 11 compares the performance of the indexing trees as a function of query tolerance. In plots (a) and (b), the horizontal axis is the query tolerance, and the vertical axis is the pruning power, where the ratio of warping window λ is 0.1 in plot (a), and 1.0 in plot (b). For all the algorithms considered, pruning power decreases with increasing query tolerance because more samples are accepted. From the two plots, the pruning power of the DCRC-Tree is higher than the others, i.e. LB_DCRC has a tighter lower bound. After querying in the indexing tree(R-Tree or DCRC-Tree), the remaining unpruned time series are sequentially scanned using the UCR suite method [27].

While searching for a given query time series on the DCRC-tree, visiting the non-leaf nodes only costs about 800 milliseconds of computation time. Therefore, the querying time cost of linear scanning is decided by the pruning power, more pruning power leads to lower time cost. Plots (c)($\lambda = 0.1$) and (d)($\lambda = 1.0$) provide a comparison of the computation time for the different algorithms. Again, DCRC-Tree outperforms the other methods. The curves are all monotonically increasing, which reflects the fact that as ε increases, more candidate data are retrieved.

Fig. 12 shows the pruning power with varying λ for different indexing structures. In plots (a) and (b), the horizontal axis is the ratio of warping window λ, and the vertical axis is the pruning power, where the tolerance $\varepsilon = 0.1$ and 1.0 in plots (a)and (b), respectively. The pruning power decreases with increasing λ, because as the warping window λ increases, the lower bound



(a) Pruning power for varying of dimension (λ = 0.1) (b) Pruning power for varying of dimension (λ = 1.0) (c) Computation Time for varying of dimension (λ = 0.1) (d) Time Consumption for varying of dimension (λ = 1.0)

Figure 13: Indexing performance comparisons with varying tree node dimension

becomes lower so that more candidates are accepted. From the two plots, it is evident that the pruning power of DCRC-Tree is greater than the other methods.

In plots (c) and (d) (the tolerance ε is set to 0.1 and 1.0, respectively), the DCRC-Tree significantly reduces the number of candidates, which greatly reduces the time complexity of indexing as only a small part of the dataset needs to be linear scanned.

Due to the computation time cost and "dimensional curse" [8, 14], the node dimension of tree-like indexing structures is usually set to 20. In Fig. 13, we compare the influence of the tree node dimension on pruning power. In Fig. 13, the horizontal axis is the node dimension which varies from 10 to 30, and the vertical axis is the pruning power, where $\varepsilon = 1.0$, $\lambda = 0.2$ in plot(a), and $\lambda = 1.0$ in plot(b), respectively. The dimensional conversion adopts the PAA algorithm [8, 14] and the unpruned results are linear scanned [27]. The results show that, as expected, the time consumption increases with increasing dimension, and that the LB_DCRC tree substantially outperforms the other methods across the full range of dimensions considered.

7 Conclusion

Dynamic time warping has become a popular approach for measuring the similarity of time series, with lower bound based techniques used to speed up its application to pruning series in search processes. This paper has presented DCRC as a novel structure for tightly covering a given set of time series under the DTW distance, and based on this structure proposed the Hierarchical DCRC (HDCRC) to generate DCRC-tree indexing. We also introduce a lower bound of the DTW distance, which is the distance between a query time series and a given DCRC-based cover set. The tightness of the lower bound, which we have proven theoretically, makes it highly suited to pruning when querying on indexing trees. With the aid of extensive experimental studies we have illustrated that LB_DCRC has more stable performance than competing methods for time series indexing.

Our future research will focus on multivariate time series, an increasingly important topic in time series data mining, with the view to extending the DCRC structure to cover the set of multivariate time series. Since multivariate time series have both variable-based and time-based dimensions, we will endeavor to explore a new way to represent multivariate time series appropriately.

Acknowledgements

The authors sincerely thank the editors and the anonymous reviewers for the very helpful and kind comments that have enhanced the presentation of our paper. The authors would also like to thank the UCR time series classification archive and Prof. Keogh for providing the datasets used in the study. This work is supported in part by the National Natural Science Foundation of China (Grant Nos. 61751205, 91746209, 61772102).

References

[1] Rakesh Agrawal, Christos Faloutsos, and Arun Swami. Efficient similarity search in sequence databases. In *Proceedings of International Conference on Foundations of Data Organization and Algorithms*, pages 69–84, Boston, MA, 1993. Springer.

[2] C. L. Philip Chen and Chun-Yang Zhang. Data-intensive applications, challenges, techniques and technologies: A survey on big data. *Information Sciences*, 275:314–347, 2014.

[3] Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping Chen, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The UCR time series classification archive, 2018.

[4] Jonathon Edstrom, Dongliang Chen, Yifu Gong, Jinhui Wang, and Na Gong. Data-pattern enabled self-recovery low-power storage system for big video data. *IEEE Transactions on Big Data*, 5(1):95–105, 2019.

[5] Philippe Esling and Carlos Agon. Time-series data mining. *ACM Computing Surveys*, 45(1):12:1–34, 2012.

[6] Tak-Chung Fu. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1):164–181, 2011.

[7] Josif Grabocka, Martin Wistuba, and Lars Schmidt-Thieme. Fast classification of univariate and multivariate time series through shapelet discovery. *Knowledge and Information Systems*, 49(2):429–454, 2016.

[8] Antonin Guttman. R-trees: A dynamic index structure for spatial searching. In *ACM Sigmod International Conference on Management of Data*, pages 47–57, New York, NY, 1984. ACM.

[9] Hong He and Yonghong Tan. Unsupervised classification of multivariate time series using VPCA and fuzzy clustering with spatial weighted matrix distance. *IEEE Transactions on Cybernetics*, 50(3):1096–1105, 2020.

[10] Jilin Hu, Bin Yang, Chenjuan Guo, and Christian S. Jensen. Risk-aware path selection with time-varying, uncertain travel costs: A time series approach. *VLDB Journal*, 27(2):179–200, 2018.

[11] Andrey Ignatov. Real-time human activity recognition from accelerometer data using convolutional neural networks. *Applied Soft Computing*, 62:915–922, 2018.

[12] Fumitada Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):67–72, 1975.

[13] Janusz Kacprzyk, Anna Wilbik, and Sawomir Zadrożny. Linguistic summarization of time series using a fuzzy quantifier driven aggregation. *Fuzzy Sets and Systems*, 159(12):1485–1499, 2008.

[14] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and Information Systems*, 7(3):358–386, 2005.

[15] Eamonn Keogh, Li Wei, Xiaopeng Xi, Michail Vlachos, Sang-Hee Lee, and Pavlos Protopapas. Supporting exact indexing of arbitrarily rotated shapes and periodic time series under Euclidean and warping distance measures. *VLDB Journal*, 18(3):611–630, 2009.

[16] Daniel Lemire. Faster retrieval with a two-pass dynamic-time-warping lower bound. *Pattern Recognition*, 42:2169–2180, 2009.

[17] Hailin Li and Libin Yang. Extensions and relationships of some existing lower-bound functions for dynamic time warping. *Journal of Intelligent Information Systems*, 43(1):59–79, 2014.

[18] Qing Li, Yan Chen, Jun Wang, Yuanzhu Chen, and Hsin Chun Chen. Web media and stock markets: A survey and future directions from a big data perspective. *IEEE Transactions on Knowledge and Data Engineering*, 30(2):381–399, 2018.

[19] Su-Chen Lin, Mi-Yen Yeh, and Ming-Syan Chen. Non-overlapping subsequence matching of stream synopses. *IEEE Transactions on Knowledge and Data Engineering*, 30(1):101–114, 2018.

[20] Mingqin Liu, Xiaoguang Zhang, and Guiyun Xu. Continuous motion classification and segmentation based on improved dynamic time warping algorithm. *International Journal of Pattern Recognition and Artificial Intelligence*, 32(2):1850002, 2018.

[21] Karl Øyvind Mikalsen, Filippo Maria Bianchi, Cristina Soguero-Ruiz, and Robert Jenssen. Time series cluster kernel for learning similarities between multivariate time series with missing data. *Pattern Recognition*, 76:569–581, 2018.

[22] Tanmoy Mondal, Nicolas Ragot, Jean-Yves Ramel, and Umapada Pal. Comparative study of conventional time series matching techniques for word spotting. *Pattern Recognition*, 73:47–64, 2018.

[23] Usue Mori, Alexander Mendiburu, and Jose A. Lozano. Similarity measure selection for clustering time series databases. *IEEE Transactions on Knowledge and Data Engineering*, 28(1):181–195, 2016.

[24] Abdullah Mueen, Nikan Chavoshi, Noor Abu-El-Rub, Hossein Hamooni, Amanda Minnich, and Jonathan MacCarthy. Speeding up dynamic time warping distance for sparse time series data. *Knowledge and Information Systems*, 54(1):237–263, 2018.

[25] Abdullah Mueen and Eamonn Keogh. Extracting optimal performance from dynamic time warping. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2129–2130, New York, NY, 2016. ACM.

[26] Sanghyun Park, Dongwon Lee, and Wesley W. Chu. Fast retrieval of similar subsequences in long sequence databases. In *Proceedings of 1999 Workshop on Knowledge and Data Engineering Exchange*, pages 60–67, Chicago, IL, 1999. IEEE.

[27] Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 262–270, New York, NY, 2012. ACM.

[28] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

[29] Yilin Shen, Yanping Chen, Eamonn Keogh, and Hongxia Jin. Accelerating time series searching with large uniform scaling. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 234–242, Bologna, Italy, 2018. SIAM.

[30] Nguyen Thanh Son and Duong Tuan Anh. Discovery of time series k-motifs based on multidimensional index. *Knowledge and Information Systems*, 46(1):59–86, 2016.

[31] Tao Sun, Hongbo Liu, Hong Yu, and C. L. Philip Chen. Degree-pruning dynamic planning approaches to central time series through minimizing dynamic time warping distance. *IEEE Transactions on Cybernetics*, 47(7):1719–1729, 2017.

[32] Chang Wei Tan, Fran?ois Petitjean, and Geoffrey Webb. Elastic bands across the path: A new framework and method to lower bound DTW. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 522–530, Alberta, Canada, 05 2019. SIAM.

[33] Chang Wei Tan, Geoffrey I. Webb, and François Petitjean. Indexing and classifying gigabytes of time series under time warping. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 282–290, Houston, TX, 2017. SIAM.

[34] Zhiyi Tan, Yanfeng Wang, Ya Zhang, and Jun Zhou. A novel time series approach for predicting the long-term popularity of online videos. *IEEE Transactions on Broadcasting*, 62(2):436–445, 2016.

[35] Jingren Tang, Hong Cheng, Yang Zhao, and Hongliang Guo. Structured dynamic time warping for continuous hand trajectory gesture recognition. *Pattern Recognition*, 80:21–31, 2018.

[36] Xindong Wu, Xingquan Zhu, Gongqing Wu, and Wei Ding. Data mining with big data. *IEEE Transactions on Knowledge and Data Engineering*, 26(1):97–107, 2014.

[37] Youxi Wu, Yao Tong, Xingquan Zhu, and Xindong Wu. NOSEP: Nonoverlapping sequence pattern mining with gap constraints. *IEEE Transactions on Cybernetics*, 48(10):2809–2822, 2018.

[38] Byoung-Kee Yi, Hosagrahar Visvesvaraya Jagadish, and Christos Faloutsos. Efficient retrieval of similar time sequences under time warping. In *Proceedings of the 14th International Conference on Data Engineering*, pages 201–208, Orlando, FL, 1998. IEEE.

[39] Mi Zhou and Man Hon Wong. Boundary-based lower-bound functions for dynamic time warping and their indexing. *Information Sciences*, 181(19):4175–4196, 2011.

[40] Kostas Zoumpatianos, Yin Lou, Ioana Ileana, Themis Palpanas, and Johannes Gehrke. Generating data series query workloads. *VLDB Journal*, 27(6):823–846, 2018.