


On the Complexity of Computing Time Series Medians Under the Move-Split-Merge Metric

Jana Holznigenkemper ✉

Fachbereich Mathematik und Informatik, Philipps-Universität Marburg, Germany

Christian Komusiewicz ✉ 

Institute of Computer Science, Friedrich-Schiller-Universität Jena, Germany

Nils Morawietz ✉

Fachbereich Mathematik und Informatik, Philipps-Universität Marburg, Germany

Bernhard Seeger ✉

Fachbereich Mathematik und Informatik, Philipps-Universität Marburg, Germany

Abstract

We initiate a study of the complexity of MSM-MEDIAN, the problem of computing a median of a set of k real-valued time series under the move-split-merge distance. This distance measure is based on three operations: moves, which may shift a data point in a time series; splits, which replace one data point in a time series by two consecutive data points of the same value; and merges, which replace two consecutive data points of equal value by a single data point of the same value. The cost of a move operation is the difference of the data point value before and after the operation, the cost of split and merge operations is defined via a given constant c .

Our main results are as follows. First, we show that MSM-MEDIAN is NP-hard and W[1]-hard with respect to k for time series with at most three distinct values. Under the Exponential Time Hypothesis (ETH) our reduction implies that a previous dynamic programming algorithm with running time $|I|^{\mathcal{O}(k)}$ [Holznigenkemper et al., Data Min. Knowl. Discov. '23] is essentially optimal. Here, $|I|$ denotes the total input size. Second, we show that MSM-MEDIAN can be solved in $2^{\mathcal{O}(d/c)} \cdot |I|^{\mathcal{O}(1)}$ time where d is the total distance of the median to the input time series.

2012 ACM Subject Classification Mathematics of computing → Time series analysis

Keywords and phrases Parameterized Complexity, Median String, Time Series, ETH

Digital Object Identifier 10.4230/LIPIcs.MFCS.2023.54

Funding *Jana Holznigenkemper*: Partially supported by the LOEWE initiative (Hesse, Germany) within the emergenCITY center.

Nils Morawietz: Supported by the DFG, project OPERAH, KO 3669/5-1.

1 Introduction

Computing an exact or approximate consensus of a set of real-valued time series is a critical task in many applications like nearest-neighbor classification and clustering of time series [1, 8, 10]. The algorithmic task in this problem is to compute for a given set X of time series some time series x which has a small distance to the members of X . Naturally, the quality of the computed consensus time series for the above-mentioned applications and the difficulty of computing such a consensus depend highly on the underlying time series distance function.

The best results in terms of quality and robustness of the computed distances is achieved by elastic distance measures [11]. Informally, when computing an elastic distance measure for two time series, each time series may be stretched or compressed. This ensures that the best-fitting parts are aligned. Elastic distance measures allow for comparison of time series of different lengths, are translation invariant and robust to temporal misalignment [9]. The high quality of elastic measures however comes at a high running time cost [11].



© Jana Holznigenkemper, Christian Komusiewicz, Nils Morawietz, and Bernhard Seeger; licensed under Creative Commons License CC-BY 4.0

48th International Symposium on Mathematical Foundations of Computer Science (MFCS 2023).

Editors: Jérôme Leroux, Sylvain Lombardy, and David Peleg; Article No. 54; pp. 54:1–54:15

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Consider for example the most popular elastic measure, the *dynamic time warping* (DTW) distance. The DTW distance of two time series of length n and m can be computed in $\mathcal{O}(nm)$ time by a simple dynamic programming (DP) algorithm. Under the SETH, a fairly common complexity-theoretic assumption, this cannot be improved to $\mathcal{O}(n^{2-\epsilon})$ time even when both time series have length n [3]. DTW-MEAN, the problem of computing a mean under the DTW distance measure, that is, a time series that minimizes the sum of squared DTW distances to the input time series, can be solved in $\mathcal{O}(n^{2k+1}2^k k)$ time [2]. Here, n is the maximum time series length and k is the number of input time series. While this running time is polynomial for a constant number of strings, it grows rapidly with k making the algorithm impractical for $k \geq 4$. DTW-MEAN is NP-hard [4], so a polynomial-time algorithm cannot be expected. Moreover, DTW-MEAN is W[1]-hard with respect to k and thus an FPT algorithm for k , that is, an algorithm with running time $f(k) \cdot n^{\mathcal{O}(1)}$ is presumably impossible [4]. Altogether, the complexity of distance and mean computation under the DTW measure are fairly well-understood by now.

DTW is by far not the only elastic distance measure. In this work, we study the *Move-Split-Merge* (MSM) distance [12]. Here, one time series is transformed into another using three types of operations. A *move* shifts one value in a time series, a *split* replaces one point of a time series by two subsequent points of the same value and a *merge* replaces two subsequent points of the same value by a single point of the same value. In contrast to DTW, the MSM distance fulfills the properties of a metric. Moreover, it achieves high accuracy in 1-NN classification tasks [9, 11]. This highly motivates an algorithmic study of distance computation and median finding problems. Initially, the distance computation for the MSM metric was found to be much slower than for DTW [11]. Recent progress showed that, on the practical side, the MSM distance computation can be improved so that it is competitive with state-of-the-art algorithms for DTW [5]. For MSM-MEDIAN, the problem of computing a time series that minimizes the sum of the MSM-distances to a given set X of time series, the running times are even better than for DTW-MEAN¹: There is a DP algorithm that solves MSM-MEDIAN in $\mathcal{O}(n^{k+3}2^k k^3)$ time [6].² As in the DTW-MEAN algorithm, this is a polynomial running time for fixed number of input sequences but the running time dependence on k is better. This running time advantage was also confirmed in an experimental evaluation [6]. Given the high quality of MSM-based classifications, it would be very desirable to push this running time advantage even further by finding better algorithms for MSM-MEDIAN. At this point, it should be noted that apart from the above-mentioned DP, nothing is known about the complexity of MSM-MEDIAN. This work aims at filling this gap. In particular, we study whether there is hope for substantial improvements over the current DP algorithm.

Our Results. We present two main results. First, we show that MSM-MEDIAN is NP-hard, W[1]-hard when parameterized by k , and cannot be solved in $f(k) \cdot |I|^{\mathcal{O}(k)}$ time for any computable function f , unless the ETH fails. Here, $|I|$ denotes the total input size of MSM-MEDIAN. These hardness results hold even if the total number of distinct values in the input time series is three. This implies that the previous DP algorithm for MSM-MEDIAN [6]

¹ It may seem inappropriate to directly compare a mean finding problem with a median finding problem. However, in the DTW-MEAN problem, the DTW distance measure is defined as a square root of a warping past cost [2]. Consequently, the DTW mean minimizes the sum of warping path costs and is thus in fact a median for this distance measure.

² Previously, the problem was called MSM-MEAN. Following a reviewer suggestion, we now use the name MSM-MEDIAN since the aim is to minimize the sum of distances instead of the sum of squared distances.

is close to optimal. Second, we show that MSM-MEDIAN can be solved in $2^{\mathcal{O}(d/c)} \cdot |I|^{\mathcal{O}(1)}$ time where c is the constant cost for a merge or split operation and d is a bound on the total distance of the median x^* to the input time series in X . Due to space constraints, proofs of statements marked with (*) are deferred to a full version.

2 Preliminaries

A *time series* of length n is a sequence $x = (x[1], \dots, x[n])$, where each *data point*, in short *point*, $x[i]$ is a real number. For $i \in [1, n]$ and $j \in [i, n]$, we denote by $x[i, j]$ the contiguous subsequence of x starting at i and ending at j . A contiguous subsequence $x[i, j]$ is an α -run if $x[\ell] = \alpha$ for each $\ell \in [i, j]$. Similarly, we call $x[i, j]$ a *binary run* if $x[\ell] \in \{0, 1\}$ for each $\ell \in [i, j]$. For a set of time series $X = \{x_1, \dots, x_k\}$, the i th point of the j th time series of X is denoted by $x_j[i]$; $V(X) = \{x[i] \mid x \in X, i \in [1, |x|]\}$ denotes the set of all values of points in the time series of X .

Move-Split-Merge Operations. We now define the MSM metric, following the notation of Stefan et al. [12], and the MSM-MEDIAN problem. The MSM metric allows three types of operations to transfer one time series into another: *move*, *split*, and *merge*. For time series $x = (x[1], \dots, x[n])$, a move transforms a point $x[i]$ into $x[i] + w$ for some $w \in \mathbb{R}$, that is, $\text{Move}_{i,w}(x) := (x[1], \dots, x[i-1], x[i] + w, x[i+1], \dots, x[n])$, with cost $\text{Cost}(\text{Move}_{i,w}) = |w|$. Informally, we say that there is a *move at point $x[i]$ to another point $x[i] + w$* . The split operation splits the i th element of x into two consecutive points. A split at point $x[i]$ is defined as $\text{Split}_i(x) := (x[1], \dots, x[i-1], x[i], x[i], x[i+1], \dots, x[n])$. A merge operation may be applied to two consecutive points of equal value. For $x[i] = x[i+1]$, it is given by $\text{Merge}_i(x) := (x[1], \dots, x[i-1], x[i+1], \dots, x[n])$. We say that $x[i]$ and $x[i+1]$ *merge*. Split and merge operations are inverse operations. Their costs are assumed to be equal and determined by a given nonnegative constant $c =: \text{Cost}(\text{Split}_i) = \text{Cost}(\text{Merge}_i)$.

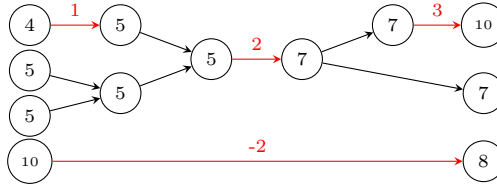
A *transformation sequence* \mathbb{S} is a tuple (S_1, \dots, S_s) with $S_j \in \{\text{Move}_{i_j, w_j}, \text{Split}_{i_j}, \text{Merge}_{i_j}\}$. A *transformation* $\mathbf{T}(x, \mathbb{S})$ of a time series x for a given transformation sequence \mathbb{S} is defined recursively via $\mathbf{T}(x, \mathbb{S}) := \mathbf{T}(S_1(x), (S_2, \dots, S_s))$ and $\mathbf{T}(x, \emptyset) := x$. We say that \mathbb{S} transforms x to y if $\mathbf{T}(x, \mathbb{S}) = y$. The cost of a transformation sequence \mathbb{S} is the sum of all individual operation costs, that is, $\text{Cost}(\mathbb{S}) := \sum_{S \in \mathbb{S}} \text{Cost}(S)$. A transformation is *optimal* if it has minimal cost. The *MSM distance* $d_{\text{MSM}(c)}(x, y)$ between two time series x and y is the cost of an optimal transformation. If c is clear from context, we may only write d_{MSM} . We let $D_{\text{MSM}(c)}(X, y) := \sum_{x \in X} d_{\text{MSM}(c)}(x, y)$ denote the distance of a sequence X of time series to a time series y . A *median* x^* of a set of time series X is a time series with minimum distance to X . The decision problem of computing a median is defined as follows.

MSM-MEDIAN

Input: A constant $c > 0$, a sequence $X := (x_1, \dots, x_k)$ of time series, and an integer d .

Question: Is there a time series x^* such that $D_{\text{MSM}(c)}(X, x^*) \leq d$?

Transformations Graphs. We further recall the concept of *transformation graphs* to describe the structure of a transformation between time series x and x^* [6, 12]. The *transformation graph* of $\mathbf{T}(x, \mathbb{S}) = x^*$ is a directed acyclic graph $G_{\mathbb{S}}(x, x^*)$ with *source nodes* $N(x) = \{u[1], \dots, u[m]\}$ and *sink nodes* $N(x^*) = \{u^*[1], \dots, u^*[n]\}$, where a node $u[i]$ represents the point $x[i]$ and a node $u^*[j]$ represents the point $x^*[j]$. The nodes which are neither source nor sink nodes are called *intermediate nodes*. All nodes have in-degree and out-degree at most 2. If there is a directed path from one node α to another node β , we say that α is *aligned*



■ **Figure 1** An optimal transformation graph between $x := (4, 5, 5, 10)$ and $y := (10, 7, 8)$ with $c = 0.1$. Move edges are red. There are two merge operations, one split operation, and move operations of total cost 8. Hence, $d(x, y) = 8.3$.

to β . The nodes of $N(x)$ that align to the same node α correspond to consecutive points in x . Each node in the node set V of $G_{\mathbb{S}}(x, x^*)$ is associated with a value given by a function $\text{val} : V \rightarrow \mathbb{R}$. For source and sink nodes we have $\text{val}(u[i]) = x[i]$ and $\text{val}(u^*[j]) = x^*[j]$. Each intermediate node is also associated with a value. The edges represent the transformation operations of \mathbb{S} . To create a transformation graph, for each operation in \mathbb{S} , a respective *move edge* or two *split* edges, or two *merge edges* are added to the graph. If a node α has outdegree 2 and is connected to a node β by a split edge and β is a child of α , then there exists a node $\gamma \neq \beta$ to which α is connected by a split edge and which is a child of α . If the nodes α and β are connected by a merge edge, α is a parent of β , and β has indegree 2, then there exists a node $\gamma \neq \alpha$ which is connected to β by a merge edge and is a parent of β . Moreover, for the split and the merge case, it holds that $\text{val}(\alpha) = \text{val}(\beta) = \text{val}(\gamma)$. A move edge can be further specified as an *increasing (inc-)* or *decreasing (dec-)* edge if the move operation adds a (not necessarily strictly) positive or negative value to the value of the parent node, respectively.

A *transformation path*, in short *path*, in $G_{\mathbb{S}}(x, x^*)$ is a directed path from a source node $u[i] \in N(x)$ to a sink node $u^*[j] \in N(x^*)$. A transformation path is *monotonic* if the move edges on this path are only inc- or only dec-edges. A transformation graph is *optimal* if it belongs to an optimal transformation. There exists an optimal transformation graph $G_{\mathbb{S}}(x, x^*)$ which can be decomposed into a sequence of distinct *trees* (T_1, \dots, T_t) with the following properties [6]: 1) The sink and source nodes of each tree T_i form a contiguous subsequence of x and y , respectively. 2) For all $i \in [1, t - 1]$, the source and sink nodes of T_i precede the source and sink nodes of T_{i+1} , respectively. 3) Each path in each T_i is monotonic. For $i \in [1, n]$ and $j \in [i, n]$, we denote by $u[i, j]$ a contiguous subsequence of nodes of x .

The *cost of a tree* T is the sum of the cost of the tree edges and denoted by $\text{Cost}_c(T)$. If the merge/split cost c is clear from the context, we may omit the subscript. If a tree contains two nodes α and β , its cost are at least $|\alpha - \beta|$. If a tree contains a split or merge edge, its cost are at least c . We call an optimal transformation graph decomposed into a sequence of trees as an *optimal transformation forest*. Figure 1 shows an example where the transformation graph consists of two trees.

► **Observation 1 (*)**. *Let X be a set of time series, let x^* be a time series, and let $i \in [1, |x^*| - 1]$. If for each time series $x \in X$, there is some optimal transformation graph containing a tree T_x such that both $u^*[i]$ and $u^*[i + 1]$ are sinks of T_x , then there is a time series y^* such that for each time series $x \in X$, $d_{\text{MSM}(c)}(x, x^*) > d_{\text{MSM}(c)}(x, y^*)$.*

Let x and y be time series of the same length. We define $d_{\text{Move}}(x, y) := \sum_{i=1}^{|x|} |x[i] - y[i]|$ as the *move distance* between x and y . The move distance describes the cost of a transformation forest between x and y that only uses move operations. Hence, $d_{\text{Move}}(x, y) = d_{\text{MSM}(c)}(x, y)$ if and only if some optimal transformation between x and y uses only move operations.

Circular Consensus String. Our negative results are obtained by a reduction from BINARY CIRCULAR CONSENSUS STRING [4]. Let s be a string of length n and let $\delta \in [1, n]$. The *circular shift of s by δ* is the string $s^{\leftarrow\delta}$ of length n with $s^{\leftarrow\delta}[i] := s[1 + (i - 1 + \delta) \bmod n]$ for each $i \in [1, n]$. We denote the *Hamming distance* of strings s_1 and s_2 by $d_{\text{Ham}}(s_1, s_2)$.

BINARY CIRCULAR CONSENSUS STRING

Input: A set $S := \{s_1, \dots, s_k\}$ of binary strings of length n and an integer d .

Question: Is there a binary string s^* of length n and a k -tuple $(\delta_1, \dots, \delta_k) \in [0, n-1]^k$ such that $\sum_{i \in [1, k]} d_{\text{Ham}}(s_i^{\leftarrow\delta_i}, s^*) \leq d$?

The Exponential Time Hypothesis (ETH) [7] implies that 3-SAT cannot be solved in $2^{o(|F|)}$ time where F is the input formula. Assuming the ETH, BINARY CIRCULAR CONSENSUS STRING cannot be solved in $f(k) \cdot n^{o(k)}$ time for any computable function f [4].

3 Finding an MSM-Median is Hard

In this section we prove our main hardness results for MSM-MEDIAN.

► **Theorem 2.** *For $c = 1$, MSM-MEDIAN is NP-hard, W[1]-hard when parameterized by k , and cannot be solved in $f(k) \cdot |I|^{o(k)}$ time for any computable function f , unless the ETH fails. This holds even if $|V(X)| = 3$.*

To show the hardness results, we present a reduction from the BINARY CIRCULAR CONSENSUS STRING-problem which is NP-hard, W[1]-hard when parameterized by k , and cannot be solved in $f(k) \cdot n^{o(k)}$ time for any computable function f , unless the ETH fails [4].

Let $I := (S, d)$ be an instance of BINARY CIRCULAR CONSENSUS STRING and let n denote the length of each binary strings of $S := \{s_1, \dots, s_k\}$ with $k := |S|$. If $d \geq n \cdot k$, then I is a trivial yes-instance. Hence, we assume that $d \leq n \cdot k$. We can assume that $k \leq n$ as otherwise I can be solved in FPT-time for k . We now describe how to construct in polynomial time an equivalent instance $I' := (c = 1, X, d')$ of MSM-MEDIAN such that $|X| = |S|$ and $|I'| \in n^{O(1)}$. Each point in each time series of X has value either 0, 1, or $A := 2d + 3$. Let $g : \{0, 1\}^* \rightarrow \{0, 1, A\}^*$ be the function where $g(0) := (0, A)$ and $g(1) := (1, A)$, and for any binary string y of length at least 2 we have $g(y) := g(y[1]) \circ \dots \circ g(y[|y|])$.

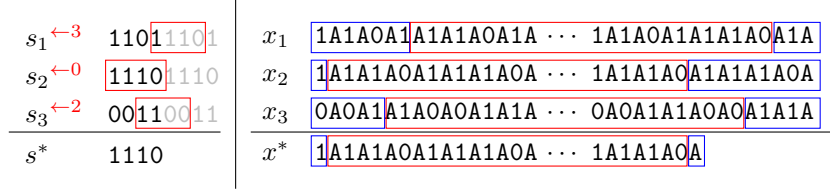
For each $i \in [1, k]$, we define a time series $x'_i := g(s_i)$. Let $R := k \cdot (n \cdot (A + 2) + 1)$. We set $X := (x_1, \dots, x_k)$, where for each $i \in [1, k]$, x_i is the concatenation of R copies of x'_i , that is, $x_i := (x'_i)^R = (g(s_i))^R = g((s_i)^R)$. Finally, we set $d' := (R - 1) \cdot d + k \cdot (n \cdot (A + 2) + 1) = (R - 1) \cdot d + k \cdot (n \cdot (2d + 5) + 1)$. This completes the construction of I' .

Note that $|X| = k$ and that $|I'| \in |I|^{O(1)} \subseteq n^{O(1)}$ since we assumed that $k \leq n$ and $d \leq n \cdot k$. Hence, to show the statement, it remains to show that I is a yes-instance of BINARY CIRCULAR CONSENSUS STRING if and only if I' is a yes-instance of MSM-MEDIAN.

I' is a yes-instance if I is a yes-instance. Let s^* be a binary string of length n and let $(\delta_1, \dots, \delta_k) \in [0, n - 1]^k$ be a k -tuple such that $\sum_{i \in [1, k]} d_{\text{Ham}}(s_i^{\leftarrow\delta_i}, s^*) \leq d$. We define a time series x' as $x' := g(s^*)$. Let x^* be the concatenation of $R - 1$ copies of x' , that is,

$$x^* := (x')^{R-1} = (g(s^*))^{R-1} = g((s^*)^{R-1}).$$

We show that $D_{\text{MSM}}(X, x^*) \leq d'$. More precisely, we show that $d_{\text{MSM}}(x_i, x^*) \leq (R - 1) \cdot d_{\text{Ham}}(s_i^{\leftarrow\delta_i}, s^*) + n \cdot (A + 2) + 1$ for each $i \in [1, k]$. Since $\sum_{s_i \in S} d_{\text{Ham}}(s_i^{\leftarrow\delta_i}, s^*) \leq d$ and $D_{\text{MSM}}(X, x^*) = \sum_{x_i \in X} d_{\text{MSM}}(x_i, x^*)$, this then implies that $D_{\text{MSM}}(X, x^*) \leq (R - 1) \cdot d + k \cdot (n \cdot (A + 2) + 1) = d'$. Informally, we obtain the bound on $d_{\text{MSM}}(x_i, x^*)$ via the



■ **Figure 2** Left: An instance of CCS for three strings s_1, s_2, s_3 with the consensus string s^* . Red rectangles show the shift δ_i , $i \in [1, 3]$. Right: Corresponding MSM-MEDIAN problem with input time series x_1, x_2 , and x_3 and the median x^* . Blue parts in the input time series align via merge-and move edges to x^* . Red parts align via move edges from the input time series to x^* .

following transformation: For the middle $(R-1) \cdot 2 \cdot n$ points of x_i , only move operations are applied. All other points at the beginning and end of each time series in X merge to the first or last point in x^* , respectively. Figure 2 shows an example.

The above-mentioned set of middle points of x_i is defined as follows. The string $(s_i)^R$ contains the substring $(s_i^{\leftarrow \delta_i})^{R-1}$ starting at index $\delta_i + 1$. Hence, $x_i = g((s_i)^R)$ contains the substring $\tilde{x}_i := g((s_i^{\leftarrow \delta_i})^{R-1}) = g(s_i^{\leftarrow \delta_i})^{R-1}$ starting at index $2 \cdot \delta_i + 1$. The time series \tilde{x}_i comprises exactly these middle points. We now bound the distance of x_i to \tilde{x}_i and the distance of \tilde{x}_i to x^* .

First, we show that $d_{\text{MSM}}(x_i, \tilde{x}_i) \leq n \cdot (A+2) + 1$. To this end, we describe a transformation graph G_i between x_i and \tilde{x}_i that consists of $(R-1) \cdot 2 \cdot n$ trees. The first tree T_1 contains the first $2 \cdot \delta_i + 1$ points of x_i as source nodes and the first point of \tilde{x}_i as the unique sink node. Similarly, the last tree $T_{(R-1) \cdot 2 \cdot n}$ contains the last $2 \cdot (n - \delta_i) + 1$ points of x_i as source nodes and the last point of \tilde{x}_i as the unique sink node. For each $\ell \in [2, (R-1) \cdot 2 \cdot n - 1]$, the tree T_ℓ consists of a single edge from the source $u_i[2 \cdot \delta_i + \ell]$ to the sink $\tilde{u}_i[\ell]$. Since x_i contains the substring \tilde{x}_i starting at index $2 \cdot \delta_i + 1$, for each $\ell \in [2, (R-1) \cdot 2 \cdot n - 1]$, we have $x_i[2 \cdot \delta_i + \ell] = \tilde{x}_i[\ell]$ which implies $\text{Cost}(T_\ell) = 0$. Hence, it remains to show that $\text{Cost}(T_1) + \text{Cost}(T_{(R-1) \cdot 2 \cdot n}) \leq n \cdot (A+2) + 1$. The following lemma upper-bounds the costs of these trees independent of the concrete binary values of their respective sources and sinks. For each $\alpha \in \{0, 1, A\}$, we use α as shortcut for the length-one time series (α) .

► **Lemma 3 (*)**. *Let y be a binary string. It holds that*

- $d_{\text{MSM}}(A \circ g(y), A) \leq |y| \cdot (A+2)$ and
- for each $\alpha_1 \in \{0, 1\}$ and each $\alpha_2 \in \{0, 1\}$, $d_{\text{MSM}}(g(y) \circ \alpha_1, \alpha_2) \leq |y| \cdot (A+2) + 1$.

Recall that sink nodes of T_1 are nodes in x^* . Since the unique sink node of T_1 has value either 0 or 1, Lemma 3 implies $\text{Cost}(T_1) \leq (\delta_i) \cdot (A+2) + 1$. Moreover, since the unique sink node of $T_{(R-1) \cdot 2 \cdot n}$ has value A , Lemma 3 implies $\text{Cost}(T_{(R-1) \cdot 2 \cdot n}) \leq (n - \delta_i) \cdot (A+2)$. Hence, $d_{\text{MSM}}(x_i, \tilde{x}_i) \leq \text{Cost}(T_1) + \text{Cost}(T_{(R-1) \cdot 2 \cdot n}) \leq n \cdot (A+2) + 1$.

Second, we show that $d_{\text{MSM}}(\tilde{x}_i, x^*) \leq (R-1) \cdot d_{\text{Ham}}(s_i^{\leftarrow \delta_i}, s^*)$. Recall that \tilde{x}_i and x^* have the same length $((R-1) \cdot 2 \cdot n)$. Hence, it is sufficient to show that $d_{\text{Move}}(\tilde{x}_i, x^*) \leq (R-1) \cdot d_{\text{Ham}}(s_i^{\leftarrow \delta_i}, s^*)$. Since $\tilde{x}_i[\ell] = x_i^*[\ell] = A$ for each even $\ell \in [1, |x^*|]$, we conclude

$$\begin{aligned}
 d_{\text{Move}}(\tilde{x}_i, x^*) &= \sum_{\text{odd } \ell \in [1, |x^*|]} |\tilde{x}_i[\ell] - x^*[\ell]| = \sum_{\ell \in [1, (R-1) \cdot n]} |\tilde{x}_i[2\ell - 1] - x^*[2\ell - 1]| \\
 &= (R-1) \cdot \sum_{\ell \in [1, n]} |\tilde{x}_i[2\ell - 1] - x^*[2\ell - 1]| \\
 &= (R-1) \cdot \sum_{\ell \in [1, n]} d_{\text{Ham}}(\tilde{x}_i[2\ell - 1], x^*[2\ell - 1])
 \end{aligned}$$

$$\begin{aligned}
 &= (R - 1) \cdot \sum_{\ell \in [1, n]} d_{\text{Ham}}(s_i^{\leftarrow \delta_i}[\ell], s^*[\ell]) \\
 &= (R - 1) \cdot d_{\text{Ham}}(s_i^{\leftarrow \delta_i}, s^*).
 \end{aligned}$$

Recall that $\tilde{x}_i = (g(s_i^{\leftarrow \delta_i}))^{R-1}$ and $x^* = (g(s^*))^{R-1}$. The second to last equation holds since by definition of g , for each $j \in [1, n]$, $g(s_i^{\leftarrow \delta_i})[2j - 1] = s_i^{\leftarrow \delta_i}[j]$ and $g(s^*)[2j - 1] = s^*[j]$. Hence, $d_{\text{MSM}}(\tilde{x}_i, x^*) \leq d_{\text{Move}}(\tilde{x}_i, x^*) = (R - 1) \cdot d_{\text{Ham}}(s_i^{\leftarrow \delta_i}, s^*)$.

Since d_{MSM} is a metric, we obtain

$$d_{\text{MSM}}(x_i, x^*) \leq d_{\text{MSM}}(x_i, \tilde{x}_i) + d_{\text{MSM}}(\tilde{x}_i, x^*) \leq (R - 1) \cdot d_{\text{Ham}}(s_i^{\leftarrow \delta_i}, s^*) + n \cdot (A + 2) + 1.$$

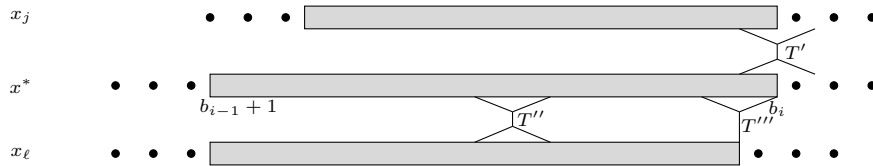
Hence, $d_{\text{MSM}}(x_i, x^*) \leq (R - 1) \cdot d_{\text{Ham}}(s_i^{\leftarrow \delta_i}, s^*) + n \cdot (A + 2) + 1$ for each time series $x_i \in X$ and thus $D_{\text{MSM}}(X, x^*) \leq d'$. Consequently, I' is a yes-instance of MSM-MEDIAN.

I' is a no-instance if I is a no-instance. If I is a no-instance, then for each binary string s^* of length n and each k -tuple $(\delta_1, \dots, \delta_k) \in [0, n - 1]^k$, $\sum_{i \in [1, k]} d_{\text{Ham}}(s_i^{\leftarrow \delta_i}, s^*) \geq d + 1$. Let x^* be a time series that minimizes $D_{\text{MSM}}(X, x^*)$. We show that $D_{\text{MSM}}(X, x^*) \geq R \cdot (d + 1) = d' + d > d$. We can assume that x^* uses only values of $V(X) = \{0, 1, A\}$ [6]. For each $i \in [1, k]$, let $G_{S_i}(x_i, x^*)$ be an optimal transformation graph between x_i and x^* . Moreover, let \mathcal{T}_i be the collection of all trees of $G_{S_i}(x_i, x^*)$. We can assume that each value in each such tree is from $V(X) = \{0, 1, A\}$ [6]. For a collection of trees \mathfrak{T} , we denote $\text{Cost}(\mathfrak{T}) := \sum_{T \in \mathfrak{T}} \text{Cost}(T)$. To show that $D_{\text{MSM}}(X, x^*) > d'$, we first introduce some notation.

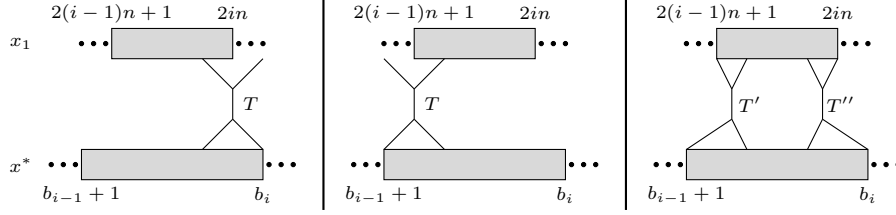
We say that a move edge (u_1, u_2) of any tree is *heavy* if $|\text{val}(u_1) - \text{val}(u_2)| > 1$. Analogously, we call path P in any tree *heavy* if at least one move edge of P is heavy. Since each node of each tree between X and x^* has a value from $\{0, 1, A\}$, a tree T contains a heavy path if and only if T contains at least one node with value A and at least one node with value either 0 or 1. In other words, if a tree T contains no heavy path, then a) the value of each node in T is A or b) the value of each node in T is from $\{0, 1\}$. Since $A = 2d + 3$, the cost of a tree with a heavy path is at least $A - 1 = 2(d + 1)$.

In the following, we take the time series $x_1 \in X$ as a pivot and regard the partial alignment of a prefix of x_1 to a prefix of x^* . Then, we analyze the cost of all other time series $x \in X \setminus \{x_1\}$ aligning to this prefix of x^* .

For each $i \in [0, R]$, let b_i be the largest number of $[0, |x^*|]$ such that $u^*[b_i]$ is the sink of a tree of \mathcal{T}_1 containing no source nodes of $u_1[2ni + 1, 2nR]$. Note that $b_0 = 0$ and that if the tree of \mathcal{T}_1 that has $u_1[1]$ as a source node also has a source node $u_1[2ni + 1]$ for some $i \in [1, R - 1]$, then $b_j = 0$ for each $j \in [0, i]$. For each $i \in [0, R]$ let \mathcal{B}_i be set of trees between any time series x of X and x^* containing only sinks of $u^*[1, b_i]$. Figure 3 depicts examples of trees belonging to \mathcal{B}_i and a tree not belonging to \mathcal{B}_i . For $i \in [1, R]$, the i th block is defined as $\mathcal{Q}_i = \mathcal{B}_i \setminus \mathcal{B}_{i-1}$. That is, each tree T in a block \mathcal{Q}_i contains only sinks of $u^*[1, b_i]$ and at least one sink of $u^*[b_{i-1} + 1, b_i]$. The i th block \mathcal{Q}_i is a *cut* if $\mathcal{Q}_i \cap \mathcal{T}_1$ has exactly the set of source nodes $u_1[2n(i - 1) + 1, 2ni]$. Figure 4 depicts two examples



■ **Figure 3** Two time series x_j and x_l with a median x^* of X ; T' is not in \mathcal{B}_i , T'' and T''' are in \mathcal{B}_i .



■ **Figure 4** The upper time series shows x_1 , the lower time series shows the median x^* . The first example is not a cut since the tree T has source nodes $u_1[j]$ with $j > 2in$. The second example is not a cut since the tree T has source nodes $u_1[j]$ with $j \leq 2(i-1)n$. The third example is a cut.

of blocks not being cuts and one example of a cut. The idea behind the definitions of cuts is as follows: If a block \mathcal{Q}_i is not a cut, then some tree $T \in \mathcal{T}_1$ with at least two sources contains $u_1[2n(i-1)]$ or $u_1[2ni+1]$ as a source. We say that \mathcal{Q}_i is a *light cut* if for each tree $T \in \mathcal{Q}_i \cap \mathcal{T}_1$, T contains no heavy path. Hence, if \mathcal{Q}_i is a light cut, then for each source node \hat{u} in $u_1[2n(i-1)+1, 2ni]$, the tree of \mathcal{T}_1 containing \hat{u} contains no heavy path. Note that a light cut \mathcal{Q}_i may still contain trees with heavy paths but these trees are not contained in \mathcal{T}_1 .

We further describe the structure of a light cut. By construction, every copy of x_1 starts with a binary value followed by an A , followed by a binary value and so on. All paths in trees of a light cut \mathcal{Q}_i are light. That is, an A in x_1 aligns to one or multiple A in x^* and a binary number in x_1 aligns to one or multiple binary numbers in x^* . That is, we have n non-empty binary runs $(r_{\text{bin}}^1, \dots, r_{\text{bin}}^n)$ and n non-empty A -runs (r_A^1, \dots, r_A^n) such that

$$x^*[b_{i-1}+1, b_i] = (r_{\text{bin}}^1 \circ r_A^1 \circ r_{\text{bin}}^2 \circ r_A^2 \circ \dots \circ r_{\text{bin}}^n \circ r_A^n).$$

We now show that each block \mathcal{Q}_i has amortized cost at least $d+1$. This implies that the total cost of the transformation forest exceeds $d' < R \cdot (d+1)$ and thus I' is a no-instance of MSM-MEDIAN. Let $i \in [1, R]$. We say that a set $J \subseteq [i, R]$ with $i \in J$ is *right-dominated* by a set $J' \subseteq [i, R]$ if $J \subseteq J'$ and $[\max(J), \max(J')] \subseteq J'$.

► **Lemma 4 (*)**. *Let $i \in [1, R]$ such that the block \mathcal{Q}_i is not a light cut. Moreover, let $J \subseteq [i, R]$ such that $i \in J$ and for each $j \in J$, the block \mathcal{Q}_j is not a light cut. Then, there is some $J' \subseteq [i, R]$ such that J is right-dominated by J' and $\text{Cost}(\cup_{j \in J'} \mathcal{Q}_j \cap \mathcal{T}_1) \geq |J'| \cdot (d+1)$.*

► **Lemma 5**. *Let I be a no-instance of BINARY CIRCULAR CONSENSUS STRING and let $i \in [0, R-1]$. If $\text{Cost}(\mathcal{B}_i) \geq i \cdot (d+1)$, then there is some $j > i$ such that $\text{Cost}(\mathcal{B}_j) \geq j \cdot (d+1)$.*

Proof. Considering the next block \mathcal{Q}_{i+1} , we distinguish whether \mathcal{Q}_{i+1} is a light cut.

Case 1: Block \mathcal{Q}_{i+1} is not a light cut. Let $J = \{i+1\}$. By Lemma 4, there is some $J' \subseteq [i+1, R]$ with $J \subseteq J'$ and $[\max(J), \max(J')] = [i+1, \max(J')] \subseteq J'$ such that $\text{Cost}(\cup_{j \in J'} \mathcal{Q}_j \cap \mathcal{T}_1) \geq |J'| \cdot (d+1)$. For $j = \max(J')$ we have $\text{Cost}(\mathcal{B}_j \setminus \mathcal{B}_i) \geq \text{Cost}(\cup_{j \in J'} \mathcal{Q}_j \cap \mathcal{T}_1) \geq |J'| \cdot (d+1) = (j-i) \cdot (d+1)$. We get

$$\begin{aligned} \text{Cost}(\mathcal{B}_j) &= \text{Cost}((\mathcal{B}_j \setminus \mathcal{B}_i) \cup \mathcal{B}_i) = \text{Cost}(\mathcal{B}_j \setminus \mathcal{B}_i) + \text{Cost}(\mathcal{B}_i) \\ &\geq (j-i) \cdot (d+1) + i \cdot (d+1) = j \cdot (d+1). \end{aligned}$$

Hence, the statement holds for j .

Case 2: Block \mathcal{Q}_{i+1} is a light cut. Recall that since \mathcal{Q}_{i+1} is a light cut, there are n non-empty binary runs $(r_{\text{bin}}^1, \dots, r_{\text{bin}}^n)$ and n non-empty A -runs (r_A^1, \dots, r_A^n) such that

$$x^*[b_i+1, b_{i+1}] = (r_{\text{bin}}^1 \circ r_A^1 \circ r_{\text{bin}}^2 \circ r_A^2 \circ \dots \circ r_{\text{bin}}^n \circ r_A^n).$$

First, we show two properties of transformation graphs from each time series of X to x^* which directly imply $\text{Cost}(\mathcal{B}_j) \geq j \cdot (d + 1)$ for some $j \geq i + 1$. Afterward, we show that transformation graphs without these properties then resembles circular shifts of the binary input strings of I . Because I is a no-instance of BINARY CIRCULAR CONSENSUS STRING, we then get $\text{Cost}(\mathcal{Q}_{i+1}) \geq d + 1$ which implies $\text{Cost}(\mathcal{B}_{i+1}) \geq (i + 1) \cdot (d + 1)$.

▷ **Claim 6 (*)**. Let $i \in [0, R - 1]$ such that \mathcal{Q}_{i+1} is a light cut. If there is some tree T in $\mathcal{B}_R \setminus \mathcal{T}_1$ that contains at least one sink node of $u^*[b_i + 1, b_{i+1}]$ and where any of the following holds:

1. T is contained in \mathcal{Q}_{i+1} and contains a heavy path or
 2. the values of the sinks of T contain at least one A and at least one binary value,
- then there is some $j > i$ such that $\text{Cost}(\mathcal{B}_j \setminus \mathcal{B}_i) \geq (j - i) \cdot (d + 1)$.

Hence, if Condition 1 or Condition 2 holds, then there is some $j \geq i + 1$ such that $\text{Cost}(\mathcal{B}_j \setminus \mathcal{B}_i) \geq (j - i) \cdot (d + 1)$. Consequently, $\text{Cost}(\mathcal{B}_j) = \text{Cost}(\mathcal{B}_j \setminus \mathcal{B}_i) + \text{Cost}(\mathcal{B}_i) \geq j \cdot (d + 1)$ which implies that the statement holds for j . In the following, we thus assume that neither Condition 1 or Condition 2 holds. This implies that

- each tree of \mathcal{Q}_{i+1} has only one source node (otherwise, Condition 1 holds) and
- each tree with at least one sink node of $x^*[b_i + 1, b_{i+1}]$ has either a) only sinks with binary values or b) only sinks with value A (otherwise, Condition 2 holds).

Note that the latter implies that each tree with at least one sink node of $x^*[b_i + 1, b_{i+1}]$ has either a) only sinks with binary values or b) only sinks with value A . Since Condition 1 does not hold, this further implies that each tree T of \mathcal{Q}_{i+1}

- has only sinks with binary values, if the value of the unique source of T is binary and
- has only sinks with value A , if the unique source of T has value A .

Next, we show that the statement holds for $j = i + 1$. To this end, we show that $\text{Cost}(\mathcal{Q}_{i+1}) \geq d + 1$. For each $q \in [1, n]$, let last^q denote the index of the last binary value of r_{bin}^q in the median. Let s^* be the length- n string such that $s^*[q] = x^*[\text{last}^q]$ for each $q \in [1, n]$.

▷ **Claim 7**. For each $p \in [1, k]$, there is some even index δ'_p such that for each $q \in [1, n]$ there is a tree T_p^q satisfying the following properties:

- (i) T_p^q is contained in $\mathcal{Q}_{i+1} \cap \mathcal{T}_p$,
- (ii) $u_p[\delta'_p + 2 \cdot q - 1]$ is the unique source node of T_p^q , and
- (iii) $u^*[\text{last}^q]$ is a sink node of T_p^q .

Proof. Let $p \in [1, k]$. We prove this statement in an inductive way. First, we show that there is an even index δ'_p such that there is a tree T_p^n satisfying Properties (i)–(iii). Afterward, we show that if for some $\ell \in [2, n]$, there is a tree T_p^ℓ satisfying Properties (i)–(iii), then the tree $T_p^{\ell-1}$ satisfies Properties (i)–(iii). This then implies the statement.

Let T_p^n be the tree of \mathcal{T}_p having $u^*[\text{last}^n]$ as a sink node and let T' be the tree of \mathcal{T}_p having $u^*[\text{last}^n + 1]$ as a sink node. Note that T_p^n satisfies Property (iii). Since last^q is the index of the last value of the n th binary run of \mathcal{Q}_{i+1} , we have $x^*[\text{last}^n + 1] = A$. Moreover, since each tree containing at least one sink node in $x^*[b_i + 1, b_{i+1}]$ has either a) only sinks with binary values or b) only sinks with value A , T_p^n and T' are distinct trees. Hence, since $\text{last}^n + 1 \leq b_{i+1}$, T_p^n is contained in \mathcal{Q}_{i+1} . This implies that T_p^n satisfies Property (i). Moreover, since no tree in \mathcal{Q}_{i+1} contains a heavy path, the values of the source nodes and the values of the sink nodes of T_p^n are all binary values. Hence, since each tree of \mathcal{Q}_{i+1} has only one source node, T_p^n has a unique source and this unique source has a binary value. Since x_p contains binary values only on odd positions, there is some even δ'_p such that $u_p[\delta'_p + 2 \cdot n - 1]$ is the unique source of T_p^n . Hence, T_p^n satisfies Properties (i)–(iii) for δ'_p .

Now assume by induction that the statement holds for T_p^ℓ . We show that the tree $T_p^{\ell-1}$ satisfies Properties (i)–(iii) as well. Since $r_A^{\ell-1}$ is a non-empty A -run and each tree of \mathcal{Q}_{i+1} has either a) only sinks with binary values or b) only sinks with value A , T_p^ℓ is not the first tree of \mathcal{T}_p . Hence, since δ'_p is even, $\delta'_p + 2 \cdot \ell - 1 \geq 3$. Let T' be the tree of \mathcal{T}_p having $u_p[\delta'_p + 2 \cdot \ell - 2]$ as a source node, and let $T_p^{\ell-1}$ be the tree of \mathcal{T}_p having $u_p[\delta'_p + 2 \cdot \ell - 3] = u_p[\delta'_p + 2 \cdot (\ell - 1) - 1]$ as a source node. Since no tree of \mathcal{Q}_{i+1} contains a heavy path, the values of all sink nodes of T_p^ℓ and $T_p^{\ell-1}$ are binary and the values of all sink nodes of T' are A . Hence, T_p^ℓ contains exactly the ℓ th binary run as sink nodes, as otherwise, T_p^ℓ contains a sink with value A or T' contains a sink with a binary value. Hence, T' contains the node of the last A of the $(\ell - 1)$ th A -run as a sink node. Since T_p^ℓ is contained in \mathcal{Q}_{i+1} , this further implies that T' is also contained in \mathcal{Q}_{i+1} . Thus, similarly to the above, T' contains exactly the $(\ell - 1)$ th A -run as sink nodes, as otherwise, T' contains a sink with binary value or $T_p^{\ell-1}$ contains a sink with value A . Hence, $T_p^{\ell-1}$ contains the node $u^*[\text{last}^{\ell-1}]$ as a sink node and thus fulfills Property (iii). Since T_p^ℓ is contained in \mathcal{Q}_{i+1} , this further implies that $T_p^{\ell-1}$ is also contained in \mathcal{Q}_{i+1} . Hence, $T_p^{\ell-1}$ fulfills Property (i). By the fact that each tree in \mathcal{Q}_{i+1} contains only one source node, this then implies that $u_p[\delta'_p + 2 \cdot (\ell - 1) - 1]$ is the unique source node of $T_p^{\ell-1}$. Hence, $T_p^{\ell-1}$ satisfies Properties (i)–(iii).

Moreover, the above proof also shows that $\delta'_p + 2 \cdot \ell - 1 \geq 3$ for each $\ell \in [2, n]$, which implies that $\delta'_p \geq 0$. Additionally, the proof also shows that for each $\ell \in [2, n]$,

- T_p^ℓ of \mathcal{T}_p contains exactly the ℓ th binary run as sink nodes and
- the tree of \mathcal{T}_p containing $u_p[\delta'_p + 2 \cdot \ell - 2]$ as unique source node, contains exactly the $(\ell - 1)$ th A -run as sink nodes.³ \triangleleft

For each $p \in [1, k]$, let δ'_p be the index fulfilling the properties of Claim 7. Moreover, for each $p \in [1, k]$ and for each $q \in [1, n]$, let T_p^q be the tree fulfilling the properties of Claim 7 with respect to δ'_p . Finally, let $\mathfrak{T} := \{T_p^q \mid p \in [1, k], q \in [1, n]\}$ denote the set of these trees. We show that $\text{Cost}(\mathfrak{T}) \geq d + 1$. Due to Property (i) of Claim 7, $\mathfrak{T} \subseteq \mathcal{Q}_{i+1}$. Hence,

$$\begin{aligned} \text{Cost}(\mathcal{B}_{i+1}) &= \text{Cost}((\mathcal{B}_{i+1} \setminus \mathcal{B}_i) \cup \mathcal{B}_i) = \text{Cost}(\mathcal{Q}_{i+1}) + \text{Cost}(\mathcal{B}_i) \\ &\geq (d + 1) + i \cdot (d + 1) \geq (i + 1) \cdot (d + 1). \end{aligned}$$

To show that $\text{Cost}(\mathfrak{T}) \geq d + 1$, we use the fact that for each binary string \hat{s} of length n and each k -tuple $(\delta_1, \dots, \delta_k)$, $\sum_{p \in [1, k]} d_{\text{Ham}}(s_p^{\leftarrow \delta_p}, \hat{s}) \geq d + 1$. In particular, this holds for s^* , the string of length n where for each index $q \in [1, n]$, $s^*[q] = x^*[\text{last}^q]$. For each $p \in [1, k]$, we set $\delta_p := \frac{\delta'_p \bmod (2n)}{2} = \frac{\delta'_p}{2} \bmod n$. Next, we show that for each $p \in [1, k]$, $\text{Cost}(\mathfrak{T}_p) \geq d_{\text{Ham}}(s_p^{\leftarrow \delta_p}, s^*)$, where $\mathfrak{T}_p := \mathfrak{T} \cap \mathcal{T}_p = \{T_p^q \mid q \in [1, n]\}$.

Let $p \in [1, k]$. Recall that $x_p = (g(s_p))^R = g((s_p)^R)$. Hence, by definition of g and $s_p^{\leftarrow \delta_p}$, for each $q \in [1, n]$,

$$\begin{aligned} s_p^{\leftarrow \delta_p}[q] &= s_p[1 + (\delta_p + q - 1) \bmod n] = (s_p)^R[\delta_p + q] \\ &= x_p[2 \cdot (\delta_p + q) - 1] = x_p[2 \cdot \delta_p + 2 \cdot q - 1] = x_p[\delta'_p + 2 \cdot q - 1]. \end{aligned}$$

Since for each $q \in [1, n]$, T_p^q contains the source $u_p[\delta'_p + 2 \cdot q - 1]$ of value $x_p[\delta'_p + 2 \cdot q - 1] = s_p^{\leftarrow \delta_p}[q]$ and the sink $u^*[\text{last}^q]$ of value $s^*[q]$, we conclude $\text{Cost}(T_p^q) \geq |s_p^{\leftarrow \delta_p}[q] - s^*[q]| \geq d_{\text{Ham}}(s_p^{\leftarrow \delta_p}[q], s^*[q])$. Hence, $\text{Cost}(\mathfrak{T}_p) = \sum_{q=1}^n \text{Cost}(T_p^q) \geq \sum_{q=1}^n d_{\text{Ham}}(s_p^{\leftarrow \delta_p}[q], s^*[q]) = d_{\text{Ham}}(s_p^{\leftarrow \delta_p}, s^*)$.

Since $\sum_{p=1}^k d_{\text{Ham}}(s_p^{\leftarrow \delta_p}, s^*) \geq d + 1$, we conclude $\text{Cost}(\mathcal{Q}_{i+1}) \geq \text{Cost}(\mathfrak{T}) \geq d + 1$. This then implies $\text{Cost}(\mathcal{B}_{i+1}) \geq (i + 1) \cdot (d + 1)$. Hence, the statement holds for $j = i + 1$. \blacktriangleleft

³ Recall that x^* minimizes $D_{\text{MSM}}(X, x^*)$ and that for each $i \in [1, k]$, G_i is a transformation graph between x_i and x^* . By Observation 1, this implies that for each $\ell \in [2, n]$, the ℓ th binary run and the $(\ell - 1)$ th A -run each have length 1. This then implies $(x^*[\text{last}^1], \dots, x^*[\text{last}^n + 1]) = g(s^*)$.

Note that $\text{Cost}(\mathcal{B}_0) = \text{Cost}(\emptyset) = 0 \geq 0 \cdot (d+1)$. Hence, due to Lemma 5, one can show via induction that $\text{Cost}(\mathcal{B}_R) \geq R \cdot (d+1)$. Since $R = k \cdot (n \cdot (A+2) + 1)$,

$$R \cdot (d+1) = (R-1) \cdot d + R + d = (R-1) \cdot d + k \cdot (n \cdot (A+2) + 1) + d = d' + d.$$

Hence $D_{\text{MSM}}(X, x^*) = \text{Cost}(\mathcal{B}_R) \geq R \cdot (d+1) > d'$ and I' is a no-instance of MSM-MEDIAN. This completes the proof of the equivalence of I and I' and thus the proof of Theorem 2. With the hardness for $c = 1$ at hand, we may also show hardness for arbitrary values of c .

► **Theorem 8 (*)**. *For every constant $c > 0$, MSM-MEDIAN is NP-hard, W[1]-hard when parameterized by k , and cannot be solved in $f(k) \cdot |I|^{o(k)}$ time for any computable function f , unless the ETH fails. This holds even if $|V(X)| = 3$.*

4 Parameterized Algorithms for MSM-Median

The algorithms presented in the following extend the DP algorithm of Holznigenkemper et al. [6]. Given a sequence X of time series of length at most n each and some $m \in \mathbb{N}$, this DP computes in time $\mathcal{O}(n^{|X|+2} \cdot 2^{|X|} \cdot |X|^2 \cdot m)$ a time series x^* of length at most m that contains only points of $V(X)$ and has minimum distance to X among all such time series.

Allowing Weights. Let $X := (x_1, \dots, x_k)$ be a sequence of time series. Moreover, let $X' := \{x_i \mid 1 \leq i \leq k\}$ be the set of time series of X and let $\omega : X' \rightarrow \mathbb{N}^+$ be the function where for each $x \in X'$, $\omega(x)$ is the number of occurrences of x in X . We call (X', ω) the *weighted equivalent* of X . We denote by $D_{\text{MSM}(c)}^\omega(X', y) := \sum_{x \in X'} (\omega(x) \cdot d_{\text{MSM}(c)}(x, y))$ the *weighted MSM-distance* between X' and y .

► **Observation 9.** *Let $X := (x_1, \dots, x_k)$ be a sequence of time series and let (X', ω) be the weighted equivalent of X . Then, for each time series x^* , $D_{\text{MSM}(c)}(X, x^*) = D_{\text{MSM}(c)}^\omega(X', x^*)$.*

► **Lemma 10.** *Let X be a set of time series of length at most n each, let $\omega : X \rightarrow \mathbb{N}^+$ be a weight function, and let $m \in \mathbb{N}$. In time $\mathcal{O}(n^{|X|+2} \cdot 2^{|X|} \cdot |X|^2 \cdot m)$ one can find a time series x^* that contains only points of $V(X)$, has length at most m , and minimizes $D_{\text{MSM}(c)}^\omega(X, x^*)$.*

Proof. We adapt the algorithm by Holznigenkemper et al. [6] by inserting the weights of the time series as follows: We fill a $(k+2)$ -dimensional table D with entries $D[\mathbf{p}, \ell, s]$, where $\mathbf{p} = (p_1, \dots, p_k)$ indicates the *current positions* of X , the index $\ell \in [1, m]$ indicates the *current position* of x^* , and s is a point in $V(X)$. The entry $D[\mathbf{p}, \ell, s]$ stores the minimal cost needed to transform the partial time series $(x_1[1, p_1], \dots, x_k[1, p_k])$ to any time series x^* of length exactly ℓ where $x^*[\ell] = s$ and x^* uses only values from $V(X)$. Since all time series are weighted, all partial time series are also weighted. Hence, the cost of all transformation operations regarding the weighted partial time series also have to be weighted.

In the DP recurrence, we distinguish two cases: Merges are applied (to the last positions of a subset of X) or splits or moves are applied (to the last positions of all time series in X). When merges are applied, the position of x^* does not change in the recurrences, this case is denoted by A_{ME} . When moves and splits are applied, then the position of x^* decreases, this case is denoted A_{MS} . In the recurrence, we consider the best of these two cases:

$$D[\mathbf{p}, \ell, s] = \min\{A_{MS}[\mathbf{p}, \ell, s], A_{ME}[\mathbf{p}, \ell, s]\}.$$

To present the recurrence for the two cases, we use index sets I_{MO} , I_{SP} , and I_{ME} representing the time series indices for which move, split, and merge operations are applied, respectively.

54:12 On the Complexity of Computing Time Series Medians Under the MSM Metric

All index sets are subsets of $[1, k]$ and $I_{MO} \cup I_{SP} = [1, k]$. For an index set I , let $\bar{\mathbf{p}}_I = (\bar{p}_1, \dots, \bar{p}_k)$ where $\bar{p}_i = p_i - 1$ for all $i \in I$ and $\bar{p}_i = p_i$ for all $i \in [1, k] \setminus I$. For merge operations, the recursive call of the function does not decrease the current position of x^* :

$$A_{ME}[\mathbf{p}, \ell, s] = \min_{I_{ME}} (D[\bar{\mathbf{p}}_{I_{ME}}, \ell, s] + \sum_{i \in I_{ME}} (\omega(x_i) \cdot C(x_i[p_i], x_i[p_i - 1], s))),$$

$$\text{where } C(u, v, w) = \begin{cases} c & \text{if } v \leq u \leq w \text{ or } v \geq u \geq w \\ c + \min(|u - v|, |u - w|) & \text{otherwise.} \end{cases}$$

For move and split operations, the recursive call of the function decreases the current position of x^* :

$$A_{MS}[\mathbf{p}, \ell, s] = \min_{s' \in V(X)} \left\{ \min_{I_{MO}, I_{SP}} (D[\bar{\mathbf{p}}_{I_{MO}}, \ell - 1, s']) + \sum_{i \in I_{MO}} (\omega(x_i) \cdot |x_i[p_i] - s|) + \sum_{i \in I_{SP}} (\omega(x_i) \cdot C(s, x_i[p_i], s')) \right\}.$$

Each single entry of A_{MS} and A_{ME} can be computed in time $2^{|X|} \cdot (|X| + n + d/c + m)^{\mathcal{O}(1)}$. For the last recursion step, the entries $D[(1, \dots, 1), 1, s]$ are computed by $D[(1, \dots, 1), 1, s] = \sum_{i=1}^k (\omega(x_i) \cdot |x_i[1] - s|)$. All entries $D[\mathbf{p}, \ell, s]$ for which $p_i < 1$ for some $i \in [1, k]$ are set to $+\infty$. If $\ell = 1$ and $p_i > 1$ for all $i \in [1, k]$, then only merge operations may be applied since the position of x^* can not be decreased anymore: $D[\mathbf{p}, 1, s] = A_{ME}[\mathbf{p}, \ell, s]$.

The minimum distance $D_{\text{MSM}(c)}^\omega(X', x^*)$ to any time series x^* of length at most m that uses only points of $V(X)$ can be computed by $\min_{\ell \in [m], s \in V(X)} (D[\mathbf{p}, \ell, s])$, where $\mathbf{p} := (|x_1|, \dots, |x_k|)$. Since the previous DP [6] runs in the desired running time and we only added weights to the DP but did not change the table structure, the running time stays the same. The corresponding time series can be found via traceback. \blacktriangleleft

► Lemma 11 (*). *Let X be a set of time series of length at most n each and let $\omega : X \rightarrow \mathbb{N}^+$. Then, each time series x^* that minimizes $D_{\text{MSM}(c)}^\omega(X, x^*)$ has length at most $n \cdot |X|$.*

Improving the Dynamic Program with a Cost-Bound. Next, we establish an intermediate FPT algorithm with the parameters $|X|$ and d/c .

► Theorem 12. *Let X be a set of time series each of length at most n , let $\omega : X \rightarrow \mathbb{N}^+$ be a weight function, let $d \in \mathbb{R}$, and let $m \in \mathbb{N}$. In time $4^{|X|} \cdot 3^{d/c} \cdot (|X| + n + d/c + m)^{\mathcal{O}(1)}$ one can find a time series x^* that contains only points of $V(X)$, has length at most m , and minimizes $D_{\text{MSM}(c)}^\omega(X, x^*)$ or correctly output that there is no such time series x^* with $D_{\text{MSM}(c)}^\omega(X, x^*) \leq d$.*

The main idea is that when given a cost budget d , we may be able to limit the number of entries in the DP-table that do not exceed a cost of d . That is, we only need to compute the entries in the DP-table that are close to the diagonal.

► Lemma 13 (*). *Let $\mathbf{p} = (p_1, \dots, p_k) \in \mathbb{N}^k$, let $\ell \in \mathbb{N}$, and let $s \in \mathbb{R}$. If $\sum_{i=1}^k |p_i - \ell| > d/c$, then $D[\mathbf{p}, \ell, s] > d$.*

We now adapt the DP-table described in the proof of Lemma 10 as follows. The table entries now have a slightly different interpretation: If the minimal cost needed to transform the partial time series $(x_1[1, p_1], \dots, x_k[1, p_k])$ to any time series x^* of length ℓ with $x^*[\ell] = s$ that only uses points from $V(X)$ is at most d , then the value of $D[\mathbf{p}, \ell, s]$ is exactly this number. Otherwise, $D[\mathbf{p}, \ell, s]$ may hold an arbitrary value larger than d , for example $d + 1$.

For each $\ell \in [1, m]$ and each $s \in V(X)$, we only compute the table entries $D[(p_1, \dots, p_k), \ell, s]$ with $\sum_{i=1}^k |p_i - \ell| \leq d/c$. Moreover, whenever a sum required to compute an entry of $D[\mathbf{p}', \ell', s']$ depends on the value of an entry $D[\mathbf{p}, \ell, s]$ with $\sum_{i=1}^k |p_i - \ell| > d/c$, then this sum is not computed but replaced by $d+1$ since the sum is at least $D[\mathbf{p}, \ell, s] > d$ as well. This is correct since each entry of D is obtained by minimizing sums of non-negative numbers. Finally, we compute $d^* := \min_{\ell \in [1, m]} \min_{s \in V(X)} D[(|x_1|, \dots, |x_k|), \ell, s]$. If $d^* > d$, we output that there is no time series x^* with the desired properties. Otherwise, we find some time series x^* with $D_{\text{MSM}(c)}^\omega(X, x^*) = d^*$ via traceback and output this time series.

To show that this modified algorithm has the running time promised in Theorem 12, we bound the number of vectors $(p_1, \dots, p_k) \in \mathbb{N}^k$ with $\sum_{i=1}^k |p_i - \ell| \leq d/c$.

► **Observation 14 (*)**. *Let $\mathbf{q} = (q_1, \dots, q_k) \in \mathbb{Z}^k$ be a vector and let $\alpha \in \mathbb{N}$. In time $\mathcal{O}(2^k \cdot 3^\alpha \cdot \alpha \cdot k)$, one can enumerate all vectors $\mathbf{p} = (p_1, \dots, p_k) \in \mathbb{Z}^k$ with $\sum_{i=1}^k |q_i - p_i| \leq \alpha$.*

By setting $\alpha = d/c$ and $q_i = \ell$ for each $i \in [1, k]$, Observation 14 implies that we have to compute at most $2^{|X|} \cdot 3^{d/c} \cdot (|X| + n + d/c + m)^{\mathcal{O}(1)}$ entries of D to compute $d^* := \min_{\ell \in [1, m]} \min_{s \in V(X)} D[(|x_1|, \dots, |x_k|), \ell, s]$. Since each entry can be computed in time $2^{|X|} \cdot (|X| + n + d/c + m)^{\mathcal{O}(1)}$, we obtain the stated running time. If $d^* \leq d$, the corresponding time series can be found via traceback in the same running time. This shows Theorem 12.

An FPT-algorithm for the Distance Bound. In this section, we now obtain an FPT algorithm for the parameter d/c , removing the running time dependence on $|X|$.

► **Theorem 15.** *MSM-MEDIAN can be solved in time $2^{\mathcal{O}(d/c)} \cdot |I|^{\mathcal{O}(1)}$. Moreover, when given a yes-instance of MSM-MEDIAN, one can find a median in the same running time.*

For a time series x , we define $X_{\text{Close}}(x) := \{y \in X \mid |y| = |x| \text{ and } d_{\text{MSM}(c)}(x, y) \leq 3 \cdot c/2\}$.

► **Lemma 16.** *Let X be a set of time series, let $x \in X$, and let x^* be any time series with $d_{\text{MSM}(c)}(x, x^*) < c/2$, then*

- $|x^*| = |x|$,
- for each $y \in X_{\text{Close}}(x)$, $d_{\text{MSM}(c)}(y, x^*) = d_{\text{Move}}(y, x^*)$, and
- for each $z \in X \setminus X_{\text{Close}}(x)$, $d_{\text{MSM}(c)}(z, x^*) \geq c$.

Proof. First, note that since $d_{\text{MSM}(c)}(x, x^*) < c/2$, each optimal transformation forest between x and x^* uses only move edges. Hence, $|x^*| = |x|$.

Next, we show that for each $y \in X_{\text{Close}}(x)$, $d_{\text{MSM}(c)}(y, x^*) = d_{\text{Move}}(y, x^*)$. By the triangle inequality, $d_{\text{MSM}(c)}(y, x^*) \leq d_{\text{MSM}(c)}(y, x) + d_{\text{MSM}(c)}(x, x^*) < 3 \cdot c/2 + c/2 = 2c$. Hence, since $|y| = |x| = |x^*|$, each transformation forest between y and x^* contains the same number of split and merge operations. Since $d_{\text{MSM}(c)}(y, x^*) < 2c$, each optimal transformation forest between y and x^* uses only move edges which implies that $d_{\text{MSM}(c)}(y, x^*) = d_{\text{Move}}(y, x^*)$.

Finally, we show that for each $z \in X \setminus X_{\text{Close}}(x)$, $d_{\text{MSM}(c)}(z, x^*) \geq c$. Let $z \in X \setminus X_{\text{Close}}(x)$. If $|z| \neq |x|$, then since $|x| = |x^*|$, each transformation forest between z and x^* contains at least one split or merge operation. Consequently, $d_{\text{MSM}(c)}(z, x^*) \geq c$. Otherwise, that is, if $|z| = |x|$, then $3 \cdot c/2 < d_{\text{MSM}(c)}(z, x)$ since $z \notin X_{\text{Close}}(x)$. By the triangle inequality, $d_{\text{MSM}(c)}(z, x) \leq d_{\text{MSM}(c)}(z, x^*) + d_{\text{MSM}(c)}(x, x^*) < d_{\text{MSM}(c)}(z, x^*) + c/2$. Hence, $3 \cdot c/2 < d_{\text{MSM}(c)}(z, x^*) + c/2$ which implies $d_{\text{MSM}(c)}(z, x^*) > c$. ◀

Proof of Theorem 15. Let $I := (X', d)$ be an instance of MSM-MEDIAN where each time series of X' has length at most n and let (X, ω) be the weighted equivalent of X' . There is a median that only uses values of $V(X)$ [6, Lemma 10]. We describe how to find in the stated running time a time series x^* with this property that minimizes $D_{\text{MSM}(c)}^\omega(X, x^*)$ or correctly detect that no such time series exists with $D_{\text{MSM}(c)}^\omega(X, x^*) \leq d$. We distinguish two cases.

Case 1: $d/c \geq |X|/2$. Hence, $|X| \leq 2 \cdot d/c$. By Lemma 11, the sought time series x^* has length at most $n \cdot |X|$. Due to Theorem 12, we can find a time series with the desired properties that has length at most $n \cdot |X|$ in time $4^{|X|} \cdot 3^{d/c} \cdot (|X| + n + d/c)^{\mathcal{O}(1)} \leq 4^{2 \cdot d/c} \cdot 3^{d/c} \cdot (|X| + n + d/c)^{\mathcal{O}(1)} = 48^{d/c} \cdot (|X| + n + d/c)^{\mathcal{O}(1)}$ or detect that no such time series exists with $D_{\text{MSM}(c)}^\omega(X, x^*) \leq d$. Hence, if such a time series x^* is found, we can correctly output that I is a yes-instance of MSM-MEDIAN and return the found time series. Otherwise, by the above, we can correctly output that I is a no-instance of MSM-MEDIAN.

Case 2: $d/c < |X|/2$. The idea is as follows: If I is a yes-instance of MSM-MEDIAN, then there is a median x^* with $D_{\text{MSM}(c)}^\omega(X, x^*) \leq d$ and some time series $\tilde{x} \in X$ with $d_{\text{MSM}(c)}(\tilde{x}, x^*) \leq d/|X| < c/2$. Lemma 16 now implies: for each time series $y \in X_{\text{Close}}(\tilde{x})$, $d_{\text{MSM}(c)}(y, x^*) = d_{\text{Move}}(y, x^*)$ and for each time series $z \in X \setminus X_{\text{Close}}(\tilde{x})$, $d_{\text{MSM}(c)}(z, x^*) \geq c$. This implies that $Z := X \setminus X_{\text{Close}}(\tilde{x})$ contains at most d/c time series. Hence, to find a median, the main algorithmic difficulty lies in finding a time series of length $|\tilde{x}|$ that uses only points of $V(X)$ and minimizes $D_{\text{MSM}(c)}^\omega(X, x^*) = D_{\text{MSM}(c)}^\omega(Z, x^*) + \sum_{y \in X_{\text{Close}}(\tilde{x})} (\omega(y) \cdot d_{\text{Move}}(y, x^*))$. We show that this can be done in time $4^{|\tilde{x}|} \cdot 3^{d/c} \cdot (|X| + n + d/c)^{\mathcal{O}(1)}$ by modifying the DP of Theorem 12. Essentially the idea is that since for each time series $y \in X_{\text{Close}}(\tilde{x})$, $d_{\text{MSM}(c)}(y, x^*) = d_{\text{Move}}(y, x^*)$, the transformation forest between y and x^* is fixed and we do not need to store current positions of time series in $X_{\text{Close}}(\tilde{x})$ as dimensions in the DP-table.

▷ **Claim 17 (*)**. Let Z be a set of time series of length at most n_Z each, let Y be a non-empty set of time series of length n_Y each, let $\omega : Z \cup Y \rightarrow \mathbb{N}^+$, and let $d \in \mathbb{R}$. In time $4^{|Z|} \cdot 3^{d/c} \cdot (|Z \cup Y| + \max(n_Z, n_Y) + d/c)^{\mathcal{O}(1)}$ one can find a time series x^* that

- contains only points of $V(Z) \cup V(Y)$, has length n_Y , and
- minimizes $d_{Z,Y}(x^*) := D_{\text{MSM}(c)}^\omega(Z, x^*) + \sum_{y \in Y} (\omega(y) \cdot d_{\text{Move}}(y, x^*))$

or correctly output that there is no such time series x^* with $d_{Z,Y}(x^*) \leq d$.

The algorithm for $d/c < |X|/2$ now works as follows: Branch into all possibilities for \tilde{x} . That is, iterate over all time series $x \in X$ and compute the sets $Y := X_{\text{Close}}(x)$ and $Z := X \setminus Y$. If $|Z| > d/c$, then continue with the next time series since x is no candidate for \tilde{x} . Otherwise, apply the algorithm behind Claim 17 for the sets Z and Y . If this algorithm returns that there is no time series x^* with the desired property, then x is not a candidate for \tilde{x} or I is a no-instance of MSM-MEDIAN. Otherwise, store the time series x^* that minimizes $D_{\text{MSM}(c)}^\omega(Z, x^*) + \sum_{y \in Y} (\omega(y) \cdot d_{\text{MSM}(c)}(y, x^*)) \leq d$. After iterating over all time series of X , output the stored time series x^* that minimizes $D_{\text{MSM}(c)}^\omega(X, x^*)$. If no such time series was found, output that I is a no-instance of MSM-MEDIAN.

By the above, this algorithm is correct. It remains to show the running time. Since for any two time series x and y , $d_{\text{MSM}(c)}(x, y)$ can be computed in $\mathcal{O}(|x| \cdot |y|)$ time [12], each individual step of this algorithm runs in $|I|^{\mathcal{O}(1)}$ time. For each time series x with $|X \setminus X_{\text{Close}}(x)| \leq d/c$, the algorithm behind Claim 17 can be applied in time $4^{|X \setminus X_{\text{Close}}(x)|} \cdot 3^{d/c} \cdot |I|^{\mathcal{O}(1)} \leq 12^{d/c} \cdot |I|^{\mathcal{O}(1)}$. Consequently, this algorithm runs in time $12^{d/c} \cdot |I|^{\mathcal{O}(1)}$.

Since in both cases the running time is at most $48^{d/c} \cdot |I|^{\mathcal{O}(1)}$, the statement holds. ◀

Finally, let us observe that the concrete value of d need not be known in advance.

► **Corollary 18.** *Let X be a sequence of time series of length at most n each, then we can find in time $2^{\mathcal{O}(d/c)} \cdot (|X| + n + d/c)^{\mathcal{O}(1)}$ a time series x^* that minimizes $D_{\text{MSM}(c)}^\omega(X, x^*) = d$.*

References

- 1 Saeed Reza Aghabozorgi, Ali Seyed Shirkhorshidi, and Ying Wah Teh. Time-series clustering - A decade review. *Inf. Syst.*, 53:16–38, 2015. doi:10.1016/j.is.2015.04.007.
- 2 Markus Brill, Till Fluschnik, Vincent Froese, Brijnesh J. Jain, Rolf Niedermeier, and David Schultz. Exact mean computation in dynamic time warping spaces. *Data Min. Knowl. Discov.*, 33(1):252–291, 2019. doi:10.1007/s10618-018-0604-8.
- 3 Karl Bringmann and Marvin Künnemann. Quadratic conditional lower bounds for string problems and dynamic time warping. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15)*, pages 79–97. IEEE Computer Society, 2015.
- 4 Laurent Bulteau, Vincent Froese, and Rolf Niedermeier. Tight hardness results for consensus problems on circular strings and time series. *SIAM J. Discret. Math.*, 34(3):1854–1883, 2020. doi:10.1137/19M1255781.
- 5 Jana Holznigenkemper, Christian Komusiewicz, and Bernhard Seeger. Exact and heuristic approaches to speeding up the MSM time series distance computation. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM '23)*, pages 451–459. SIAM, 2023. doi:10.1137/1.9781611977653.ch51.
- 6 Jana Holznigenkemper, Christian Komusiewicz, and Bernhard Seeger. On computing exact means of time series using the move-split-merge metric. *Data Min. Knowl. Discov.*, 37(2):595–626, 2023.
- 7 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.
- 8 Weiwei Jiang. Time series classification: Nearest neighbor versus deep learning models. *SN Appl. Sci.*, 2(4):721, 2020. doi:10.1007/s42452-020-2506-9.
- 9 Jason Lines and Anthony Bagnall. Time series classification with ensembles of elastic distance measures. *Data Min. Knowl. Discov.*, 29:565–592, 2015. doi:10.1007/s10618-014-0361-2.
- 10 John Paparrizos and Luis Gravano. Fast and accurate time-series clustering. *ACM Trans. Database Syst.*, 42(2):8:1–8:49, 2017. doi:10.1145/3044711.
- 11 John Paparrizos, Chunwei Liu, Aaron J. Elmore, and Michael J. Franklin. Debunking four long-standing misconceptions of time-series distance measures. In *Proceedings of the 2020 International Conference on Management of Data (SIGMOD '20)*, pages 1887–1905. ACM, 2020. doi:10.1145/3318464.3389760.
- 12 Alexandra Stefan, Vassilis Athitsos, and Gautam Das. The move-split-merge metric for time series. *IEEE Trans. Knowl. Data Eng.*, 25(6):1425–1438, 2013. doi:10.1109/TKDE.2012.88.