[Array 15 (2022) 100240](https://doi.org/10.1016/j.array.2022.100240)



Contents lists available at [ScienceDirect](https://www.elsevier.com/locate/array)

Array

journal homepage: [www.elsevier.com/locate/array](http://www.elsevier.com/locate/array)

Efficient alignment-based average delay time estimation in fluctuating delayed propagation

Atsuyoshi Nakamura [∗](#_bookmark0), Tatsuya Hayashi

*Graduate School of Information Science and Technology, Hokkaido University, Japan*

A R T I C L E I N F O A B S T R A C T

*Keywords:*

Alignment

Time delay estimation Dynamic time warping Dynamic programming

We propose an alignment-based average delay time estimation algorithm between two time series in the propagation of time-varying delay. Though the number of the minimum cost alignments may be exponential in the length of the time series, the proposed algorithm takes account of all such alignments, and as a post- alignment process, it runs in time linear in the number of the nodes in the *minimum cost alignment graph*, which is at most the length squared. The efficiency of our algorithm is confirmed through numerical experiments compared to the naive enumeration algorithm using recursive calls to traverse the graph.

# Introduction

An alignment between strings or time series is the position cor- respondence between them and the minimum-cost alignment for an application-dependent cost function is used in many areas including bioinformatics and signal processing.

Though the purpose of an alignment is different depending on its application, here we consider using it for average delay time estima- tion between time series. For two time series, the difference between corresponding positions in an alignment can be seen as an estimated delay of one time series from the other time series, so an alignment can be used to estimate the average delay time between them in the propagation of time-varying delays.

Time delay estimation among signals has been studied well in the fields of sonar and radar systems, seismology, geophysics, etc. [[1](#_bookmark18)]. In most studies of those fields, constant delay for a moment is assumed and the cross-correlation method [[2](#_bookmark19)] is most widely used for the estimation. Improvement using more realistic models [[3](#_bookmark20),[4](#_bookmark21)] and spatial prediction technique [[5](#_bookmark22)] has been done since then.

For the time-varying time delay estimation, a method using a kind of alignment, which is called DTW (Dynamic Time Warping) [[6](#_bookmark23)], has been already proposed in the study area of seismology [[7](#_bookmark24)]. No consideration, however, has been done yet on how to efficiently calculate average time delay in the case with a large number of the minimum cost alignment paths.

The minimum-cost alignment is unique with high probability when continuous values can be taken in time series, however, there might be many those alignments when finite values only can be taken in them. In such a case, what we can do to estimate the average delay time is

to calculate the delay time averaged over all the aligned positions in all the minimum-cost alignments, which we call the *mean time delay by the minimum-cost alignments*. You can enumerate all the minimum-cost alignments and calculate their delay time at each aligned position one by one, then average them. Unfortunately, this strategy is inefficient in the worst case because the number of the minimum-cost alignments can be exponential in the length of the time series.

In this paper, we propose a method to calculate mean delay time by

vertices in the *minimum-cost alignment path graph*, in addition to *𝑂*(*𝑇* 2) the minimum-cost alignments in time and space linear in the number of time and space needed for length-*𝑇* time series alignment. In the graph

that is composed of cells and minimum-cost edges between them in the alignment cost table for dynamic programming, the minimum-cost alignment path graph is the subgraph induced by the minimum-cost alignment paths. Since each aligned position in the cost table corre- sponds to a diagonal edge on the path, delay time for each edge is added only once by multiplying the number of the minimum-cost align- ment paths that pass through the edge in our method. Our numerical experiments confirm computational efficiency and estimation accuracy of average delay time by our method.

There are two major ways of aligning two sequences so as to match the corresponding positions. One is gap insertion, which is used in bioinformatics [[8](#_bookmark25)], and the other is DTW, which is used in speech recognition [[9](#_bookmark26)]. For both ways, the minimum cost alignments are known to be calculated efficiently using dynamic programming. We show the calculation method of our mean delay time for each way considering their difference.

∗ Corresponding author.

*E-mail addresses:* [atsu@ist.hokudai.ac.jp](mailto:atsu@ist.hokudai.ac.jp) (A. Nakamura), [thayashi@ist.hokudai.ac.jp](mailto:thayashi@ist.hokudai.ac.jp) (T. Hayashi).

<https://doi.org/10.1016/j.array.2022.100240>

Received 12 February 2022; Received in revised form 19 July 2022; Accepted 21 July 2022

Available online 26 July 2022

2590-0056/© 2022 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY-NC-ND license ([http://creativecommons.org/licenses/by-](http://creativecommons.org/licenses/by-nc-nd/4.0/) [nc-nd/4.0/](http://creativecommons.org/licenses/by-nc-nd/4.0/)).

This paper is organized as follows. In Section [2](#_bookmark1), we define mean Define cost function *𝑤* as *𝑤*(*𝑥, 𝑦*) = *𝑥* − *𝑦* and shift functions *𝜋*

and *𝜋*

delay time by minimum-cost alignments and show examples of its as calculation for gap-based and warping-based costs. Its efficient cal-

| | 1 2

culation way for the warping-based cost is proposed with its process for our introduced example, and the time and space complexities are analysed in Section [3](#_bookmark5). In Section [4](#_bookmark9), we show the efficiency of our calculation way by numerical experiments compared with the naive calculation way. We conclude by summarizing the paper and describing its future direction in Section [5](#_bookmark14). Calculation way for the gap-based cost is explained in [Appendix](#_bookmark16) to clarify the slight difference in calculations between the two types of costs.

# Mean delay time by minimum-cost alignments

Notation [*𝑛*] for any natural number *𝑛* denotes the set {1*,* 2*,* … *, 𝑛*}. Let *𝑌* be a subset of R. For *𝑖* = 1*,* 2, let **𝐬***𝑖* denote the time series of length *𝑇* whose *𝑡*th value is *𝑠𝑖*[*𝑡*] ∈ *𝑌* , that is, **𝐬***𝑖* = *𝑠𝑖*[1] ⋯ *𝑠𝑖*[*𝑇* ]. Let

*𝛱* denote the set of *shift function pairs* which is defined as the set of

strictly increasing function pairs (*𝜋*1*, 𝜋*2) from [*𝑇* ] to [2*𝑇* − 1] for which

*𝜋*1([*𝑇* ]) ∪ *𝜋*2([*𝑇* ]) is a set of contiguous natural numbers starting from 1,

that is,

*𝛱* = {(*𝜋 , 𝜋* ) ∣*𝜋* (1) *<* ⋯ *< 𝜋* (*𝑇* ) (*𝑖* = 1*,* 2)*,*

1

2

*𝑖*

*𝑖*

*𝜋*1([*𝑇* ]) ∪ *𝜋*2([*𝑇* ]) = [max(*𝜋*1([*𝑇* ]) ∪ *𝜋*2([*𝑇* ]))]}*.*

We let *𝛱*1 denote the subset of *𝛱* that is composed of pairs (*𝜋*1*, 𝜋*2) satisfying *𝜋*1(1) = *𝜋*2(1) = 1. An *alignment* between **𝐬**1 and **𝐬**2 defined by a shift function pair (*𝜋*1*, 𝜋*2) ∈ *𝛱* is the position correspondence in which

*𝑠*1[*𝜋*−1(*𝑘*)] corresponds to *𝑠*2[*𝜋*−1(*𝑘*)] for *𝑘* ∈ *𝜋*1([*𝑇* ]) ∩ *𝜋*2([*𝑇* ]), where *𝜋*−1



(*𝜋*1(1)*,* … *, 𝜋*1(10)) =(1*,* 3*,* 4*,* 5*,* 6*,* 7*,* 9*,* 10*,* 11*,* 12) and

(*𝜋*2(1)*,* … *, 𝜋*2(10)) =(1*,* 2*,* 3*,* 4*,* 5*,* 7*,* 8*,* 9*,* 10*,* 11)*.*

Then, the alignment between **𝐬**1 and **𝐬**2 defined by the shift function pairs (*𝜋*1*, 𝜋*2) is one of the minimum warping-based cost alignment with cost 2. (See the following table.)

*𝑘* 1 2 3 4 5 6 7 8 9 10 11 12

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *𝑠*1[*𝜋*−1(*𝑘*)]  1  *𝑠*2[*𝜋*−1(*𝑘*)]  2  *𝑠*′ [*𝑘*]  1  *𝑠*′ [*𝑘*]  2  *𝑤*(*𝑠*′ [*𝑘*]*, 𝑠*′ [*𝑘*])  1 2  *𝜋*−1(*𝑘*)  1  *𝜋*−1(*𝑘*)  2  *𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*)  2 1 | 1 |  | 1 | 0 | −1 | −1 | 1 |  | 1 | 2 | 0 | −1 |
| 0 | 1 | 1 | 0 | −1 |  | 1 | 1 | 1 | 2 | 0 |  |
| 1 | 1 | 1 | 0 | −1 | −1 | 1 | 1 | 1 | 2 | 0 | −1 |
| 0 | 1 | 1 | 0 | −1 | −1 | 1 | 1 | 1 | 2 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 |  | 2 | 3 | 4 | 5 | 6 |  | 7 | 8 | 9 | 10 |
| 1 | 2 | 3 | 4 | 5 |  | 6 | 7 | 8 | 9 | 10 |  |
| 0 |  | 1 | 1 | 1 |  | 0 |  | 1 | 1 | 1 |  |

For this alignment, I(*𝜋*1*, 𝜋*2) = {3*,* 4*,* 5*,* 7*,* 9*,* 10*,* 11} and ( *𝑘*∈I(*𝜋 ,𝜋* )

∑ 1 2

(*𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*))*,* I(*𝜋 , 𝜋* ) ) = (6*,* 7). Considering following variations,

2  1  | 1 2 |



1 2 *𝑖*

is the inverse function of *𝜋𝑖*.



based and gap-based. Consider a cost function between values *𝑤* ∶ There are mainly two types of alignment cost functions, warping- (*𝑌* ∪ {␣}) × (*𝑌* ∪ {␣}) → R, where ␣ is the special value corresponding to a gap. Then, *alignment cost 𝑆*(**𝐬**1*,* **𝐬**2*,* (*𝜋*1*, 𝜋*2)) is defined by



*𝑆*(**𝐬**1*,* **𝐬**2*,* (*𝜋*1*, 𝜋*2)) = ∑ *𝑤*(*𝑠*′ [*𝑘*]*, 𝑠*′ [*𝑘*])*.*

∑

there are 20 alignments that achieve the minimum cost 2 and

Here, for *𝑖* = 1*,* 2,

*𝑠*′[*𝑘*] = *𝑠𝑖*[*𝜋*−1(*𝑘*)]

*𝑖*

1 2

*𝑘*∈*𝜋* ([*𝑇* ])∪*𝜋* ([*𝑇* ])

1

2

( (*𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*))*,* I(*𝜋 , 𝜋* ) ) for them are (4*,* 5) for 6 align- ments, (4*,* 6) for 5 alignments, (4*,* 7) for 1 alignment, (5*,* 6) for 5 align-

ments, (5*,* 7) for 2 alignments and (6*,* 7) for 1 alignment. Thus, the mean delay time of **𝐬**2 from **𝐬**1 by the minimum-cost alignments is

| |

*𝑘*∈I(*𝜋*1 *,𝜋*2 )

2

1

1

2

in warping-based cost, where *𝜋*−1(*𝑘*) = max{*ℎ* ∣ *𝜋𝑖*(*ℎ*) ≤ *𝑘*}, and

′[ ] = {*𝑠𝑖*[*𝜋*−1(*𝑘*)] (*𝑘* ∈ *𝜋𝑖*([*𝑇* ]))

4×6 + 4×5 + 4×1 + 5×5 + 5×2 + 6×1

5×6 + 6×5 + 7×1 + 6×5 + 7×2 + 7×1

=

89 118

≈ 0*.*754*.*

*𝑠𝑖 𝑘*

*𝑖*

␣ (otherwise)*,*

**Example 2.** Let *𝑌* = {0*,* 1} and consider sequences **𝐬**1 = 001000100 and

in gap-based cost. Note that *𝜋*−1(*𝑘*) = *𝜋*−1(*𝑘*) for *𝑘* ∈ *𝜋*([*𝑇* ]). Then, the

*minimum alignment cost* between **𝐬**1 and **𝐬**2 is min(*𝜋*1 *,𝜋*2 )∈*𝛱*1 *𝑆*(**𝐬**1*,* **𝐬**2*,* (*𝜋*1*,*

*𝜋*2)) for warping-based cost and min(*𝜋*1 *,𝜋*2 )∈*𝛱 𝑆*(**𝐬**1*,* **𝐬**2*,* (*𝜋*1*, 𝜋*2)) for gap- based cost. Let *𝛱* ∗(**𝐬**1*,* **𝐬**2) be the set of all the minimum-cost alignments

(*𝜋 , 𝜋* ) between **𝐬**

and **𝐬** . Then, *mean delay time of* **𝐬**

*from* **𝐬**

*by the*

**𝐬**2 = 000100010. For *𝛼* ≥ 2, we consider alignments of time series **𝐬**1 and

**𝐬**2 using symmetric cost function *𝑤*(*𝑥, 𝑦*) defined as follows:

⎧⎪0 ((*𝑥, 𝑦*) = (0*,* 0)*,* (1*,* 1))

1 ((*𝑥, 𝑦*) = (0*,* ␣)*,* (␣*,* 0))

1 2 1 2

2 1 *𝑤*(*𝑥, 𝑦*) = ⎪⎨

(1)

*minimum-cost alignments* is defined as

∑(*𝜋 ,𝜋* )∈*𝛱* ∗(**𝐬** *,***𝐬** ) ∑*𝑘*∈I(*𝜋 ,𝜋* )(*𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*))

1 2 1 2 1 2 2 1 *,*

∑(*𝜋*1 *,𝜋*2 )∈*𝛱* ∗(**𝐬**1 *,***𝐬**2 ) |I(*𝜋*1*, 𝜋*2)|

where I(*𝜋*1*, 𝜋*2) is the set of aligned positions which is defined to be

*𝜋*1([*𝑇* ]) ∩ *𝜋*2([*𝑇* ]) ⧵ {1} for warping-based cost and *𝜋*1([*𝑇* ]) ∩ *𝜋*2([*𝑇* ]) for

gap-based cost. In the next section, we propose an efficient algorithm

for calculating the mean delay time by the minimum-cost alignments.

**Example 1.** Let *𝑌* = R and consider sequences

(*𝑠*1[1]*,* … *, 𝑠*1[10]) =(1*,* 1*,* 0*,* −1*,* −1*,* 1*,* 1*,* 2*,* 0*,* −1) and

(*𝑠*2[1]*,* … *, 𝑠*2[10]) =(0*,* 1*,* 1*,* 0*,* −1*,* 1*,* 1*,* 1*,* 2*,* 0)*.*

*𝛼* ((*𝑥, 𝑦*) = (0*,* 1)*,* (1*,* 0))

∞ ((*𝑥, 𝑦*) = (1*,* ␣)*,* (␣*,* 1)(␣*,* ␣))*.*

⎪⎩

In the alignment using this cost function, each value 1 in one se- quence is strongly preferred to be aligned to value 1 in the other

(2 × (position difference) *> 𝛼*) or the number of letters 1 is different. sequence by shifting positions unless their position difference is large

Consider the case with *𝛼* = 3. Then, the minimum gap-based alignment cost is 2 and there are 6 alignments whose alignment costs are the minimum. One of the minimum cost alignments between **𝐬**1 and

**𝐬**2 is defined by shift functions (*𝜋*1(1)*,* … *, 𝜋*1(9)) = (2*,* 3*,* 4*,* 5*,* 6*,* 7*,* 8*,* 9*,* 10)

and (*𝜋*2(1)*,* … *, 𝜋*2(9)) = (1*,* 2*,* 3*,* 4*,* 5*,* 6*,* 7*,* 8*,* 10). (See the following table.)

*𝑘* 1 2 3 4 5 6 7 8 9 10

Then, all the paths from (1*,* 1) to (*𝑇 , 𝑇* ) on *𝐺* correspond to the minimum cost alignments. We call this graph *𝐺*(*𝑉 , 𝐸*) the *alignment path graph*

*between* **𝐬**1 *and* **𝐬𝟐**. We also call the induced subgraph of *𝐺* by all the

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *𝑠*1[*𝜋*−1(*𝑘*)]  1  *𝑠*2[*𝜋*−1(*𝑘*)]  2  *𝑠*′ [*𝑘*]  1  *𝑠*′ [*𝑘*]  2  *𝑤*(*𝑠*′ [*𝑘*]*, 𝑠*′ [*𝑘*])  1 2  *𝜋*−1(*𝑘*)  1  *𝜋*−1(*𝑘*)  2  *𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*)  2 1 |  | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |  | 0 |
| ␣ | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | ␣ | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |  | 9 |
|  | 1 | 1 | 1 | 1 | 1 | 1 | 1 |  | 0 |

paths corresponding to the minimum cost alignments, the *minimum cost*

*alignment path graph between* **𝐬**1 *and* **𝐬𝟐**, and let *𝐺*∗(*𝑉* ∗*, 𝐸*∗) denote it.

**Example 3.** *𝐷* for time series **𝐬**1 and **𝐬**2 with cost function *𝑤*(*𝑥, 𝑦*) =

– *𝑦* in [Example](#_bookmark4) [1](#_bookmark4) and its corresponding graph *𝐺* are shown in [Fig.](#_bookmark7) [1](#_bookmark7).

|*𝑥* |

For this alignment,

I(*𝜋*1*, 𝜋*2) = {2*,* 3*,* 4*,* 5*,* 6*,* 7*,* 8*,* 10} and

)

The 20 minimum cost alignments correspond to the paths from (1*,* 1) to

(10*,* 10) on *𝐺*.

To calculate the mean delay time of **𝐬**2 from **𝐬**1 by the minimum-

∑

cost alignments, it is enough to calculate two values, (*𝜋 ,𝜋* )∈*𝛱*∗(**𝐬** *,***𝐬** )

( ∑ ∑

| |

(*𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*))*,* I(*𝜋 , 𝜋* )

*𝑘*∈I(*𝜋*1 *,𝜋*2 )

. Each mini-

(*𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*)) and ∑

1 2 1 2

I(*𝜋 , 𝜋* )

1

2

2

= (7*,* 8)*.*

*𝑘*∈I(*𝜋*1 *,𝜋*2 )

Similarly,

1 | 1 2 |

mum-cost alignment (*𝜋*1*, 𝜋*2) can be represented by a path

(*𝜋*−1(1)*, 𝜋*−1(1))*,* (*𝜋*−1(2)*, 𝜋*−1(2))*,* … *,* (*𝜋*−1(*𝑇* ′)*, 𝜋*−1(*𝑇* ′))

2

1

(*𝜋*1 *,𝜋*2 )∈*𝛱* ∗(**𝐬**1 *,***𝐬**2 )

1 2

1 2

1 2

( ∑ −1 −1 |)

(*𝜋*2 (*𝑘*) − *𝜋*1 (*𝑘*))*,* I(*𝜋*1*, 𝜋*2)

′

1

2

1

2

|

*𝑘*∈I(*𝜋*1 *,𝜋*2 )

with (*𝜋*−1(1)*, 𝜋*−1(1)) = (1*,* 1) and (*𝜋*−1(*𝑇* ′)*, 𝜋*−1(*𝑇* ′)) = (*𝑇 , 𝑇* ) in *𝐺*, where

*𝑇*

= max(*𝜋*1(*𝑇* ) ∪ *𝜋*2(*𝑇* )). Furthermore, for *𝑘* ≥ 2,

for the other best alignments are

*𝑘* ∈ I(*𝜋*1*, 𝜋*2) ⇔ *𝜋*−1(*𝑘* − 1) + 1 = *𝜋*−1(*𝑘*) for *𝑖* = 1*,* 2

*𝑖 𝑖*

(5*,* 8)*,* (6*,* 8)*,* (6*,* 8)*,* (7*,* 8)*,* (8*,* 8)*,*

so the mean delay time of **𝐬**2 from **𝐬**1 by the minimum-cost alignments

is

5 + 6 × 2 + 7 × 2 + 8 39

holds. In *𝐺*, let *𝐵*(*𝑡*1*, 𝑡*2) be the number of paths from (*𝑡*1*, 𝑡*2) to (*𝑇 , 𝑇* ) and let *𝐹* (*𝑡*1*, 𝑡*2) be the number of paths from (1*,* 1) to (*𝑡*1*, 𝑡*2). Then, the num- ber of the minimum-cost alignments (*𝜋*1*, 𝜋*2) that contains edge ((*𝑡*1 −

1*, 𝑡*2 − 1)*,* (*𝑡*1*, 𝑡*2)) is calculated as *𝐹* (*𝑡*1 − 1*, 𝑡*2 − 1)*𝐵*(*𝑡*1*, 𝑡*2). Using the above

1

2

1 2

1

2

1 2

1

2

2

8 × 6 = 48 = 0*.*8125*.*

facts, ∑(*𝜋 ,𝜋* )∈*𝛱* ∗(**𝐬** *,***𝐬**|I)(*𝜋*1*, 𝜋*2)| and ∑(*𝜋 ,𝜋* )∈*𝛱* ∗(**𝐬** *,***𝐬** ) ∑*𝑘*∈I(*𝜋 ,𝜋* ()*𝜋*−1(*𝑘*) −

**3. Efficient calculation**

The calculation of mean delay time by the minimum cost alignments between two sequences is a time-consuming task when there are many

minimum cost alignments. As a solution to this issue, we propose a fast

1

(*𝜋*1 *,𝜋*2 )∈∑*𝛱* ∗(**𝐬**1 *,***𝐬**2 )

| 1 2 |

*𝜋*−1(*𝑘*))

can be calculated as

= ∑

((*𝑡*1 −1*,𝑡*2 −1)*,*(*𝑡*1 *,𝑡*2 ))∈*𝐸*∗

I(*𝜋 , 𝜋* )

*𝐹* (*𝑡*1 − 1*, 𝑡*2 − 1)*𝐵*(*𝑡*1*, 𝑡*2) and

algorithm for this task. In this section, we explain the way of calculating

the mean delay time for the warping-based cost. See [Appendix](#_bookmark16) with

respect to the way of calculating the mean delay time for the gap-based cost.

First, review the popular calculation algorithm for the minimum cost alignment using dynamic programming. Consider the alignment

for two time series **𝐬**1 = *𝑠*1[1] ⋯ *𝑠*1[*𝑇* ] and **𝐬**2 = *𝑠*2[1] ⋯ *𝑠*2[*𝑇* ]. De-

note *𝐷*(*𝑡*1*, 𝑡*2) be the minimum alignment cost between *𝑠*1[1] ⋯ *𝑠*1[*𝑡*1]

and *𝑠*2[1] ⋯ *𝑠*2[*𝑡*2]. Then, *𝐷*(*𝑡*1*, 𝑡*2) can be represented as the following

recursive formula.

*𝐷*(*𝑡*1*, 𝑡*2)

∑ ∑ (*𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*))

= (*𝑡*2 − *𝑡*1)*𝐹* (*𝑡*1 − 1*, 𝑡*2 − 1)*𝐵*(*𝑡*1*, 𝑡*2)*.*

∑ ∗

(*𝜋 ,𝜋* )∈*𝛱* ∗(**𝐬** *,***𝐬** ) *𝑘*∈I(*𝜋 ,𝜋* )

2

1

1

2

1 2

1

2

((*𝑡*1 −1*,𝑡*2 −1)*,*(*𝑡*1 *,𝑡*2 ))∈*𝐸*

*𝐵*(*𝑡*1*, 𝑡*2) can be represented as the following recursive formula:

*𝐵*(*𝑡*1*, 𝑡*2)

1 ((*𝑡*1*, 𝑡*2) = (*𝑇 , 𝑇* ))

⎧⎪

1{((*𝑡*1*, 𝑡*2)*,* (*𝑡*1*, 𝑡*2 +1)) ∈ *𝐸*}*𝐵*(*𝑡*1*, 𝑡*2 +1) (*𝑡*1 = *𝑇 , 𝑡*2 *<𝑇* )

⎪

1{((*𝑡 , 𝑡* )*,* (*𝑡* +1*, 𝑡* )) ∈ *𝐸*}*𝐵*(*𝑡* +1*, 𝑡* ) (*𝑡 <𝑇 , 𝑡* = *𝑇* )=

⎪ 1 2 1 2 1 2 1 2

⎪

⎧*𝑤*(*𝑠* [1]*, 𝑠* [1]) (*𝑡* = *𝑡* = 1)

1

2

1

2

⎨⎪1{((*𝑡*1*, 𝑡*2)*,* (*𝑡*1*, 𝑡*2 +1)) ∈ *𝐸*}*𝐵*(*𝑡*1*, 𝑡*2 +1)

⎪*𝐷*(*𝑡 , 𝑡* − 1) + *𝑤*(*𝑠* [1]*, 𝑠* [*𝑡* ]) (*𝑡* = 1*, 𝑡 >* 1)

1

2

1

2

2

1

2

+1{((*𝑡*1*, 𝑡*2)*,* (*𝑡*1 +1*, 𝑡*2)) ∈ *𝐸*}*𝐵*(*𝑡*1 +1*, 𝑡*2)

= ⎪⎨*𝐷*(*𝑡* − 1*, 𝑡* ) + *𝑤*(*𝑠* [*𝑡* ]*, 𝑠* [1]) (*𝑡 >* 1*, 𝑡* = 1)

1

2

1

1

2

1

2

⎪⎩ +1{((*𝑡*1*, 𝑡*2)*,* (*𝑡*1 +1*, 𝑡*2 +1)) ∈ *𝐸*}*𝐵*(*𝑡*1 +1*, 𝑡*2 +1) (*𝑡*1*, 𝑡*2 *<𝑇* )*,*

min{*𝐷*(*𝑡*1 − 1*, 𝑡*2)*, 𝐷*(*𝑡*1*, 𝑡*2 − 1)*, 𝐷*(*𝑡*1 − 1*, 𝑡*2 − 1)}

⎪

⎪⎩

0 otherwise. *𝐵*(*𝑡*1*, 𝑡*2) for the vertex (*𝑡*1*, 𝑡*2) in *𝑉* ∗ can be obtained by starting from *𝐵*(*𝑇 , 𝑇* ) and calculating *𝐵*(*𝑡*1*, 𝑡*2) in reverse lexicographic

where 1{⋅} is an indicator function, that is, 1{⋅} = 1 if ‘⋅’ holds and

+*𝑤*(*𝑠*1[*𝑡*1]*, 𝑠*2[*𝑡*2]) (*𝑡*1*, 𝑡*2 *>* 1)*.*

*𝐷*(*𝑇 , 𝑇* ) is the minimum alignment cost between **𝐬**1 and **𝐬**2, and *𝐷*(*𝑇 , 𝑇* ) can be calculated by calculating *𝐷*(*𝑡*1*, 𝑡*2) in the order of (*𝑡*1*, 𝑡*2) = (1*,* 1)*,* … *,* (1*, 𝑇* )*,* (2*,* 1)*,* … *,* (2*, 𝑇* ), ⋯ *,* (*𝑇 ,* 1)*,* … *,* (*𝑇 , 𝑇* ) using the above re-

cursive formula.

Consider the directed graph *𝐺* = (*𝑉 , 𝐸*) with

*𝑉* ={(*𝑡*1*, 𝑡*2) ∣ *𝑡*1*, 𝑡*2 ∈ {1*,* … *, 𝑇* }}

*𝐸* ={((*𝑡*1*, 𝑡*2 − 1)*,* (*𝑡*1*, 𝑡*2)) ∣

order of (*𝑡*1*, 𝑡*2) using this recursive formula.

*𝐹* (*𝑡*1*, 𝑡*2) can be also expressed by recursive formula as follows:

*𝐹* (*𝑡*1*, 𝑡*2)

1 ((*𝑡*1*, 𝑡*2) = (1*,* 1))

⎧⎪

1{((*𝑡*1*, 𝑡*2 −1)*,* (*𝑡*1*, 𝑡*2)) ∈ *𝐸*}*𝐹* (*𝑡*1*, 𝑡*2 −1) (*𝑡*1 = 1*, 𝑡*2 *>* 1)

⎪

1{((*𝑡*

– 1*, 𝑡* )*,* (*𝑡 , 𝑡* )) ∈ *𝐸*}*𝐹* (*𝑡* −1*, 𝑡* ) (*𝑡 >* 1*, 𝑡* = 1)

1{((*𝑡*1*, 𝑡*2 −1)*,* (*𝑡*1*, 𝑡*2)) ∈ *𝐸*}*𝐹* (*𝑡*1*, 𝑡*2 −1)

*𝐷*(*𝑡 , 𝑡* ) = *𝐷*(*𝑡 , 𝑡*

1

2

1

2

– 1) + *𝑤*(*𝑠* [*𝑡* ]*, 𝑠* [*𝑡* ])}

= ⎪⎨ 1

2 1 2 1 2 1 2

∪ {((*𝑡*1 − 1*, 𝑡*2)*,* (*𝑡*1*, 𝑡*2)) ∣

1

1

2

2

*𝐷*(*𝑡*1*, 𝑡*2) = *𝐷*(*𝑡*1 − 1*, 𝑡*2) + *𝑤*(*𝑠*1[*𝑡*1]*, 𝑠*2[*𝑡*2])}

∪ {((*𝑡*1 − 1*, 𝑡*2 − 1)*,* (*𝑡*1*, 𝑡*2)) ∣

*𝐷*(*𝑡*1*, 𝑡*2) = *𝐷*(*𝑡*1 − 1*, 𝑡*2 − 1) + *𝑤*(*𝑠*1[*𝑡*1]*, 𝑠*2[*𝑡*2])}*.*

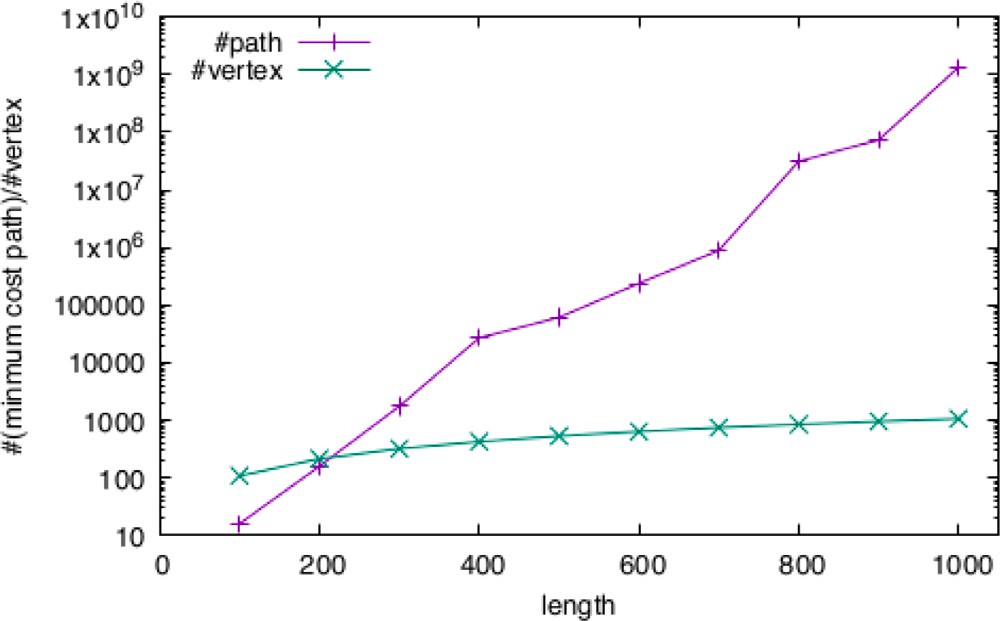
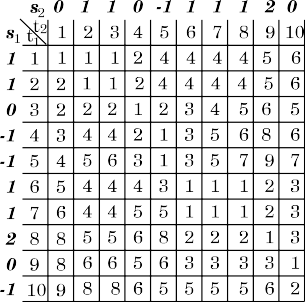
+1{((*𝑡*1 −1*, 𝑡*2)*,* (*𝑡*1*, 𝑡*2)) ∈ *𝐸*}*𝐹* (*𝑡*1 −1*, 𝑡*2)

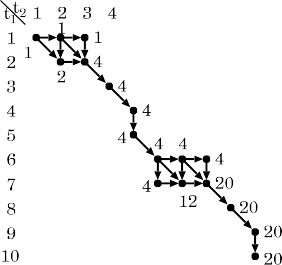
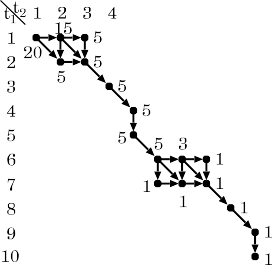
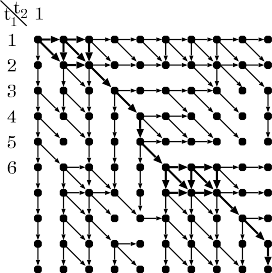
+1{((*𝑡*1 −1*, 𝑡*2 −1)*,* (*𝑡*1*, 𝑡*2)) ∈ *𝐸*}*𝐹* (*𝑡*1 −1*, 𝑡*2 −1) (*𝑡*1*, 𝑡*2 *>* 1)*,*

⎪⎪⎩

*𝐹* (*𝑡*1*, 𝑡*2) for the vertex (*𝑡*1*, 𝑡*2) in *𝑉* ∗ can be obtained by calculating

*𝐹* (*𝑡*1*, 𝑡*2) in lexicographic order of (*𝑡*1*, 𝑡*2) using this recursive formula.

sequence pairs with length= 100*,* … *,* 1000 in the minimum cost alignment path graphs. **/ig. 2.** The numbers of the minimum-cost paths and vertices averaged over 100



where

*𝑟*1*,*1*,* … *, 𝑟*1*,𝑇 , 𝑟*2*,*1*, 𝑟*2*,*2 ∼ uniform distribution over

{−50*,* −49*,* … *,* 49*,* 50}*,*



**/ig. 1.** *𝐷* for time series **𝐬**

and **𝐬**

with cost function *𝑤*(*𝑥, 𝑦*) = *𝑥* − *𝑦* in [Example](#_bookmark4) [1](#_bookmark4),

*𝑛*1*,* … *, 𝑛𝑇* ∼ uniform distribution over

1

its corresponding graph

2 | |

and on the minimum cost paths. The directed

*𝐺*, and *𝐵*

*𝐹*

{−50*𝜎,* −50*𝜎* − 1 … *,* 50*𝜎* − 1*,* 50*𝜎*}*,*

edges in the paths from (1*,* 1) to (10*,* 10) on *𝐺* are bolded. *𝐵*(*𝑡*1 *, 𝑡*2 )s and *𝐹* (*𝑡*1 *, 𝑡*2 )s for

(*𝑡*1*, 𝑡*2) only in the paths corresponding to the minimum cost alignments are shown and

*𝐵*(*𝑡*1 *, 𝑡*2 )s and *𝐹* (*𝑡*1 *, 𝑡*2 )s for other (*𝑡*1 *, 𝑡*2 ) are 0 and not needed to be calculated.

**Example 4.** *𝐵*(⋅*,* ⋅) and *𝐹* (⋅*,* ⋅) for time series **𝐬**1 and **𝐬**2 with cost function

*𝑤*(*𝑥, 𝑦*) = *𝑥* − *𝑦* in [Example](#_bookmark4) [1](#_bookmark4) is shown in [Fig.](#_bookmark7) [1](#_bookmark7). From the values in B and F, ∑∑(*𝜋*1 *,𝜋*2 )∈*𝛱* ∗(**𝐬**1 *,***𝐬**2 ) |I(*𝜋*1*, 𝜋*2)| = 2×1×5+3×4×5+2×4×1+2×20×1 =

| |

118 and

(*𝜋 ,𝜋* )∈*𝛱* ∗(**𝐬** *,***𝐬** ) ∑*𝑘*∈I(*𝜋 ,𝜋* )(*𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*)) = 0 × 1 × 5 + 1 × 1 ×

1 (*𝑡* = 2 & *𝛥* = 1)

2 (*𝑡* = 3 & *𝛥* = 2)

⎧⎪

⎪⎨⎪

*𝛥𝑡* = *𝛥𝑡*−1 with prob. 0.9 ((*𝑡* = 3 & *𝛥* = 1) or *𝑡 >* 3) 1 + (*𝛥𝑡*−1%2) with prob. 0.1 ((*𝑡* = 3 & *𝛥* = 1) or *𝑡 >* 3)

⎪⎩

{

*𝛥* = 1 with prob. 0.9

2 with prob. 0.1

(2)

5 + 2 × 1 × 4 × 5 + 0 × 4 × 5 + 0 × 4 × 1 + 1 × 4 × 1 + 2 × 1 × 20 × 1 = 89. Thus, the

1

2

1 2

1

2

2

1

mean delay time by the minimum cost alignments is 89∕118 ≈ 0*.*754,

which coincides with the calculation in [Example](#_bookmark4) [1](#_bookmark4).

Table *𝐷* and graph *𝐺*, which are non-original parts, can be con- structed in *𝑂*(*𝑇* 2) time and space. The rest process is the original part for calculating the mean delay time of **𝐬**2 from **𝐬**1 by the mini- mum cost alignments. The construction of tables *𝐵* and *𝐹* is done in

| | ∑(*𝜋*1 *,𝜋*2 )∈*𝛱* ∗(**𝐬**1 *,***𝐬**2 ) | 1 2 |

*𝑂*( *𝑉* ∗ ) time and space. Calculation of I(*𝜋 , 𝜋* ) and

∑(*𝜋 ,𝜋* )∈*𝛱* ∗(**𝐬** *,***𝐬** ) ∑*𝑘*∈I(*𝜋 ,𝜋* )(*𝜋*−1(*𝑘*) − *𝜋*−1(*𝑘*)) using tables *𝐵* and *𝐹* is also

| | | |

1

2

1 ∗2

1

2

2

1

done in *𝑂*( *𝑉*

) time and space. So totally, the rest part specific to our

Sequence **𝐬**1 is a random sequence in which *𝑠*1[1] = 0 and the difference

set of integers between −50 and 50. Random variable *𝛥*, which takes 1 between two consecutive terms follows uniform distribution over the with probability 0.9 and 2 with probability 0.1, decides whether *𝑠*1[1] propagates to **𝐬**2 at *𝑡* = 2 or *𝑡* = 3, that is, *𝑠*2[2] = *𝑠*1[1] if *𝛥* = 1 and

*𝑠*2[3] = *𝑠*1[1] if *𝛥* = 2. For *𝑡* ≤ *𝛥*, *𝑠*2[1] and *𝑠*2[*𝑡*] − *𝑠*2[*𝑡* − 1] is generated

according to uniform distribution over {−50*,* … *,* 50}. For *𝑡 > 𝛥*, *𝑠*2[*𝑡*] takes *𝑠*1[*𝑡* − *𝛥𝑡*], where delay time *𝛥𝑡* is 1 or 2, and *𝛥𝑡* takes the same value as *𝛥𝑡*−1 with probability 0.9 and a different value with probability

0.1. The number of the minimum-cost paths and vertices averaged over

100 sequence pairs for each length= 100*,* … *,* 1000 in the minimum-cost

task can be calculated in *𝑂*( *𝑉* ∗ ) time and space.

The naive calculation using the recursive formula for *𝐵* can be

realized by recursive calls, but its time complexity linearly depends on

to *𝑉* . the number of the minimum cost alignments, which can be exponential

∗

| |

# Numerical experiment

* 1. *Data generation*

We generate 100 integer sequence pairs (**𝐬**1*,* **𝐬**2) for each length *𝑇* = 100*,* 200*,* … *,* 1000 and noise scale *𝜎* = 0 (without noise) as follows. Note

that operator % is the modulus operator.

*𝑠* [*𝑡*] = 0 (*𝑡* = 1)

1 {

*𝑠*1[*𝑡* − 1] + *𝑟*1*,𝑡* (*𝑡* ≥ 2)

⎧⎪*𝑟*2*,𝑡* (*𝑡* = 1)

alignment path graphs is plotted in [Fig.](#_bookmark6) [2](#_bookmark6). You can see that the number of vertices increases linearly, but the number of the minimum-cost paths increases exponentially.

We also generate 100 noisy length-1000 sequence pairs by setting

noise scale *𝜎* = 0*.*1*,* 0*.*2*,* … *,* 1*.*0 to check the estimation accuracy of

average delay time.

* 1. *Experimental setting*

Algorithms are implemented by C++ language and executed in Mac Pro (Late 2013) (CPU: 8-Core Intel(R) Xeon(R) E5-1680 v2 @ 3.00 GHz, Memory: 64 GB).

* 1. *Results*

In the calculation of mean delay time by all the minimum-cost paths, we compare the computational efficiency of our proposed met- hod with that of a naive depth-first search method. The naive depth-

*𝑠*2[*𝑡*] =

⎨ 2

*𝑠* [*𝑡* − *𝛥* ] + *𝑛*

*𝑠* [*𝑡* − 1] + *𝑟*

⎪

(otherwise)

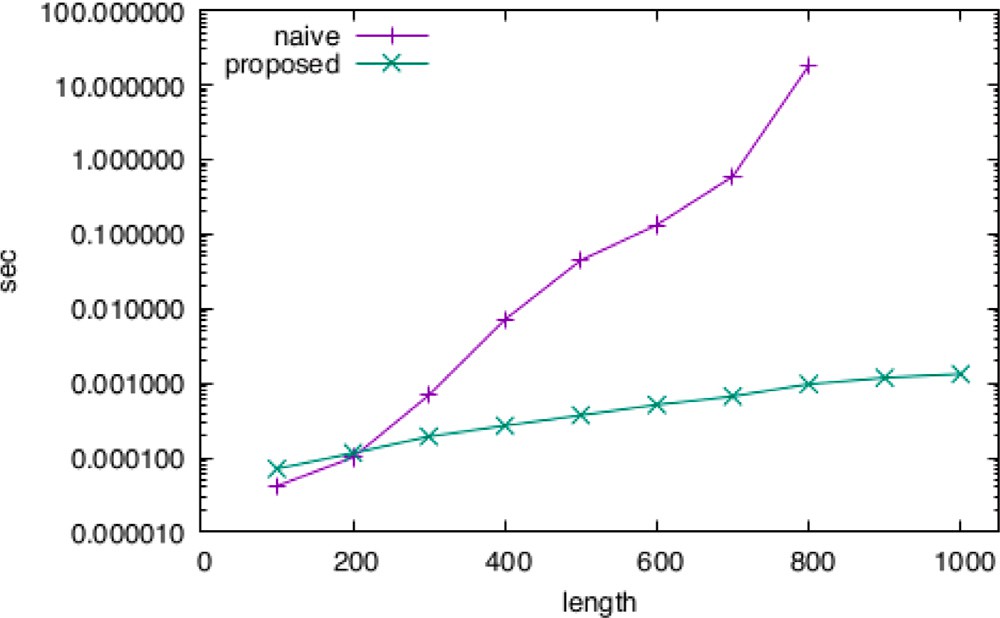
2*,𝑡*

(*𝑡* = 2 & *𝛥* = 2)

ing the minimum-cost alignment path graph from vertex (*𝑇 , 𝑇* ) in a first method searches all the minimum-cost alignment paths by travers-

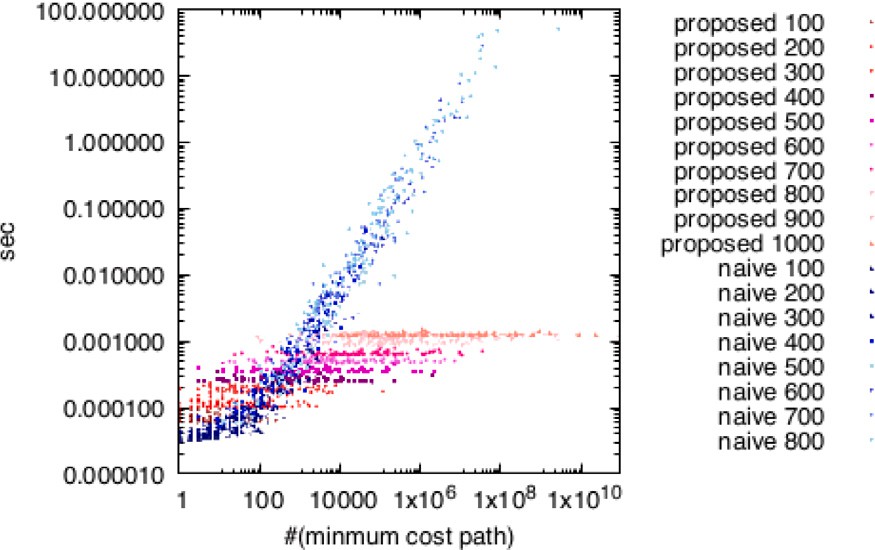
⎩ 1 *𝑡 𝑡*

depth-first manner.

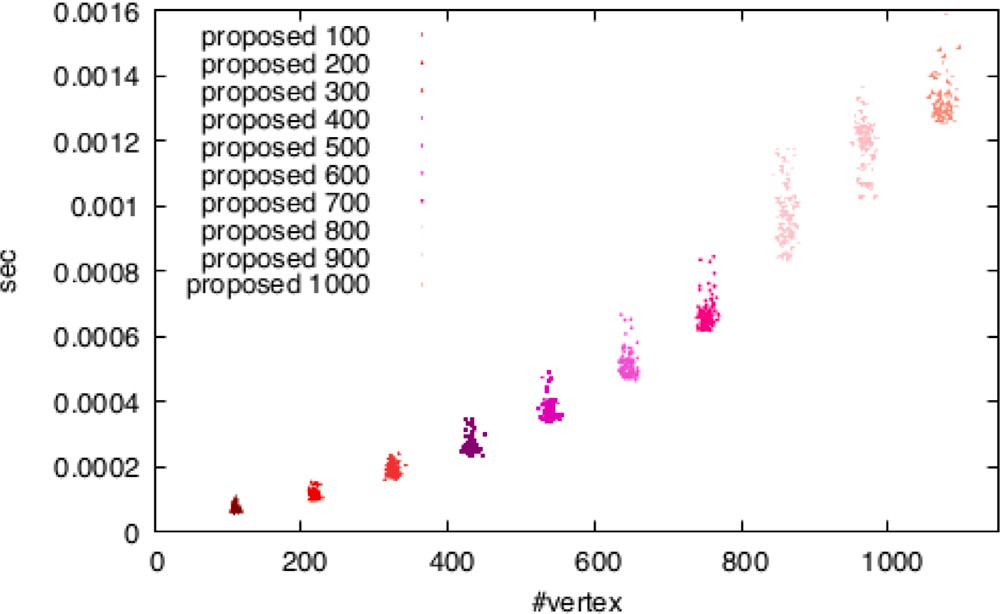


naive algorithms for sequence pairs with length= 100*,* … *,* 1000. Each average processing **/ig. 3.** Average processing time of calculating average delay time by proposed and

length-900 and length-1000 sequence pairs for the naive algorithm due to its longness. time is averaged over 100 sequence pairs. We gave up measuring processing time for

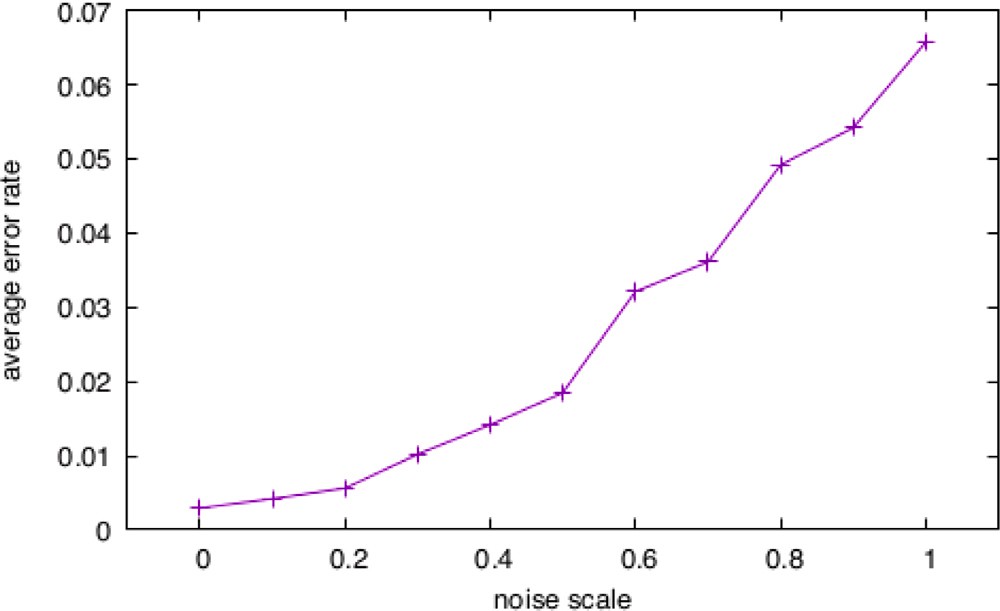


**/ig. 4.** Scatter (#(minimum-cost paths), processing time) plot of the proposed and naive algorithms.



**/ig. 5.** Scatter (#(vertex number in the minimum-cost alignment path graph), processing time) plot of the proposed algorithm.

The result is shown in [Fig.](#_bookmark10) [3](#_bookmark10). The average processing time of our proposed algorithm increases close to linearly while that of the naive algorithm increases exponentially. The relation between them looks similar to the relation between the numbers of vertices and minimum- cost alignment paths. In fact, the processing time of the naive algorithm linearly increases as a function of the number of the minimum-cost alignment paths though that number does not affect the processing time

**/ig. 6.** Average error rate of the proposed algorithm for noise scale *𝜎* = 0*.*1*,* … *,* 1*.*0.

of the proposed algorithm, as shown in the scatter plot of [Fig.](#_bookmark11) [4](#_bookmark11). A close to linear dependency on the number of vertices can be confirmed for the proposed algorithm by the scatter plot of [Fig.](#_bookmark12) [5](#_bookmark12).

Finally, we checked the accuracy of the average delay time esti- mated by our proposed method. The ground truth of average delay time

is defined as *𝛥𝑡* generated by Eq. ([2](#_bookmark8)) averaged over *𝑡* = *𝛥* + 1*,* … *, 𝑇* .

Its error rate is defined as the absolute difference between estimated

and true average delay time divided by true average delay time. Error rates averaged over 100 sequence pairs with length 1000 for noise scale

*𝜎* = 0*,* 0*.*1*,* 0*.*2*,* … *,* 1*.*0 are shown in [Fig.](#_bookmark13) [6](#_bookmark13). You can see that the error rates

is the same as the fluctuation width (*𝜎* = 1*.*0). are very small and the rate is less than 7% even when the noise width

# Conclusions and future work

The proposed alignment-based average delay time estimation al- gorithm is efficient in the case with a large number of minimum cost alignments; its computation time, excluding the alignment time, increases linearly in the number of nodes in the *minimum cost alignment path graph*, which is at most the time series length squared while the number of the minimum cost alignments may be exponential in the length. In our numerical experiments, the number of the minimum cost

alignments between 101-value-range length-1000 time series is more

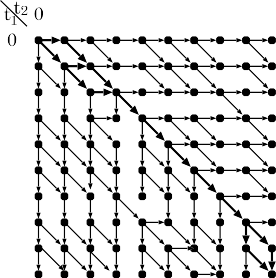
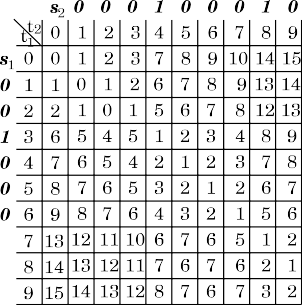
than a billion on average, which indicates that we cannot ignore the issue of a large number of the minimum cost alignments in some applications. Our method can be also applied to time series whose delay time fluctuates intensely, such as stock price and sales histories, so it might be interesting to use it for analyses of such time series in data mining. In fact, our technique developed in this paper was used in the edge set estimation of a propagation graph in applications[1](#_bookmark15) of stock price and cells’ firing analyses [[10](#_bookmark27)]. We are now considering the possibility of applying our method to various other fields.

# CRediT authorship contribution statement

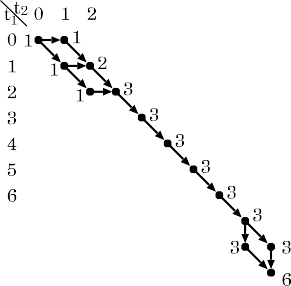
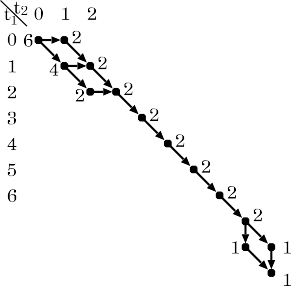
**Atsuyoshi Nakamura:** Conceptualization, Methodology, Software, Writing. **Tatsuya Hayashi:** Methodology, Software.

1 Paper [[10](#_bookmark27)] focuses on how to estimate directed edge in a propagation graph using time delay sum (mean delay time that is not divided by the number of aligned positions), and no description is there on how to efficiently calculate it.

# Declaration of competing interest



One or more of the authors of this paper have disclosed potential or pertinent conflicts of interest, which may include receipt of payment, either direct or indirect, institutional support, or association with an entity in the biomedical field which may be perceived to have poten- tial conflict of interest with this work. Atsuyoshi Nakamura reports financial support was provided by Japan Society for the Promotion of Science.







# Acknowledgements

We would like to thank Prof. Kazuki Horikawa and Prof. Tamiki Komatsuzaki for motivating us to study this research. This work was supported by JSPS KAKENHI Grant Number JP18H05413, Japan.

# Appendix. Calculation of mean delay time for the gap-based cost



In the alignment between two sequences **𝐬**1 and **𝐬**2, either **𝐬**1 or **𝐬**2 must not be a null time series for the warping-based cost, but both **𝐬**1 and **𝐬**2 can be null time series for the gap-based cost. Thus, the min- imum alignment cost *𝐷*(*𝑡*1*, 𝑡*2) between *𝑠*1[1] ⋯ *𝑠*1[*𝑡*1] and *𝑠*1[1] ⋯ *𝑠*1[*𝑡*2] for *𝑡*1 = 0 or *𝑡*2 = 0 is needed to be calculated, where *𝑠*1[1] ⋯ *𝑠*1[0] represents the null time series. The recursive formula of *𝐷*(*𝑡*1*, 𝑡*2) for

the gap-based cost is the following:

*𝐷*(*𝑡*1*, 𝑡*2)

⎧⎪0 (*𝑡*1 = *𝑡*2 = 0)

**/ig. A.7.** *𝐷* for time series **𝐬**1 and **𝐬**2 with cost function ([1](#_bookmark3)) setting *𝛼* = 3 in [Example](#_bookmark2) [2](#_bookmark2), its corresponding graph *𝐺*, and *𝐵* and *𝐹* on the minimum cost paths. The directed edges in the paths from (0*,* 0) to (9*,* 9) on *𝐺* are bolded. *𝐵*(*𝑡*1 *, 𝑡*2 )s and *𝐹* (*𝑡*1 *, 𝑡*2 )s for (*𝑡*1*, 𝑡*2) only in the paths corresponding to the minimum cost alignments are needed to

be calculated.

**Example 5.** *𝐷* for time series **𝐬**1 and **𝐬**2 with cost function ([1](#_bookmark3)) setting

*𝛼* = 3 in [Example](#_bookmark2) [2](#_bookmark2), its corresponding graph *𝐺*, the number *𝐵*(*𝑡*1*, 𝑡*2) of paths from (*𝑡*1*, 𝑡*2) to (9*,* 9) and the number *𝐹* (*𝑡*1*, 𝑡*2) of paths from (0*,* 0)

to ( )

*𝑡*1*, 𝑡*2

on the minimum cost paths in *𝐺* are shown in [Fig.](#_bookmark17) [A.7](#_bookmark17). The

*𝐷*(*𝑡*1*, 𝑡*2 − 1) + *𝑤*(␣*, 𝑐*2[*𝑡*2]) (*𝑡*1 = 0*, 𝑡*2 *>* 0)

⎪

*𝐷*(*𝑡*

– 1*, 𝑡* ) + *𝑤*(*𝑐* [*𝑡* ]*,* ␣) (*𝑡*

*>* 0*, 𝑡*

= 0)

⎪⎨

minimum cost alignments correspond to the paths from (0*,* 0) to (9*,* 9)

= 1 2 1 1

⎪ ⎧⎪ *𝐷*(*𝑡*1 − 1*, 𝑡*2) + *𝑤*(*𝑐*1[*𝑡*1]*,* ␣) ⎫⎪

1 2 in *𝐺*. From the values in B and F, we can calculate the mean delay time

by the minimum cost alignments as

min *𝐷*(*𝑡 , 𝑡* − 1) + *𝑤*(␣*, 𝑐* [*𝑡* ])

⎪ ⎨ 1 2 2 2

⎬

⎪⎩ ⎪ *𝐷*(*𝑡* − 1*, 𝑡* − 1) + *𝑤*(*𝑐* [*𝑡* ]*, 𝑐* [*𝑡* ])

⎩ 1 2 1 1 2 2 ⎭

(*𝑡*1*, 𝑡*2 *>* 0)*.*

⎪

0 × 1 × 4 + 0 × 1 × 2 + 1 × 1 × 2 + 1 × 2 × 2

+5 × 1 × 3 × 2 + 1 × 3 × 1 + 0 × 3 × 1 39

The directed graph *𝐺*(*𝑉 , 𝐸*) whose paths represent the minimum cost

alignments can be constructed as

*𝑉* ={(*𝑡*1*, 𝑡*2) ∣ *𝑡*1*, 𝑡*2 ∈ {0*,* 1*,* … *, 𝑇* }}

*𝐸* = {((*𝑡*1*, 𝑡*2 − 1)*,* (*𝑡*1*, 𝑡*2)) ∣ *𝐷*(*𝑡*1*, 𝑡*2) = *𝐷*(*𝑡*1*, 𝑡*2 − 1) + *𝑤*(␣*, 𝑐*2[*𝑡*2])}

∪{((*𝑡*1 − 1*, 𝑡*2)*,* (*𝑡*1*, 𝑡*2)) ∣ *𝐷*(*𝑡*1*, 𝑡*2) = *𝐷*(*𝑡*1 − 1*, 𝑡*2) + *𝑤*(*𝑐*1[*𝑡*1]*,* ␣)}

∪{((*𝑡*1 − 1*, 𝑡*2 − 1)*,* (*𝑡*1*, 𝑡*2)) ∣ *𝐷*(*𝑡*1*, 𝑡*2) = *𝐷*(*𝑡*1 − 1*, 𝑡*2) + *𝑤*(*𝑐*1[*𝑡*1]*, 𝑐*2[*𝑡*2])}*.*

All the paths from (0*,* 0) to (*𝑇 , 𝑇* ) on *𝐺* correspond to the minimum cost alignments. The number *𝐵*(*𝑡*1*, 𝑡*2) of paths from (*𝑡*1*, 𝑡*2) to (*𝑇 , 𝑇* )

warping-based cost. The recursive formula of *𝐹* (*𝑡*1*, 𝑡*2) is can be represented by the same recursive formula as that for the

*𝐹* (*𝑡*1*, 𝑡*2)

1 ((*𝑡*1*, 𝑡*2) = (0*,* 0))

⎧⎪

1{((*𝑡*1*, 𝑡*2 −1)*,* (*𝑡*1*, 𝑡*2)) ∈ *𝐸*}*𝐹* (*𝑡*1*, 𝑡*2 −1) (*𝑡*1 = 0*, 𝑡*2 *>* 0)

⎪

1{((*𝑡*

– 1*, 𝑡* )*,* (*𝑡 , 𝑡* )) ∈ *𝐸*}*𝐹* (*𝑡* −1*, 𝑡* ) (*𝑡 >* 0*, 𝑡* = 0)

1 × 4 + 2 × 1 × 2 + 2 × 2 + 5 × 3 × 2 + 3 × 1 + 3 × 1 = 48 = 0*.*8125*,*

which coincides with the calculation in [Example](#_bookmark2) [2](#_bookmark2).

# References

1. [Chen J, Huang Y, Benesty J. Time delay estimation. In: Huang Y, Benesty J,](http://refhub.elsevier.com/S2590-0056(22)00075-3/sb1) [editors. Audio signal processing for next-generation multimedia communication](http://refhub.elsevier.com/S2590-0056(22)00075-3/sb1) [systems. Norwell, MA: Kluwer; 2004, p. 197–227.](http://refhub.elsevier.com/S2590-0056(22)00075-3/sb1)
2. Knapp C, Carter G. The generalized correlation method for estimation of time delay. IEEE Trans Acoust Speech Signal Process 1976;24(4):320–7. [http://dx.doi.](http://dx.doi.org/10.1109/TASSP.1976.1162830) [org/10.1109/TASSP.1976.1162830](http://dx.doi.org/10.1109/TASSP.1976.1162830).
3. Manickam T, Vaccaro R, Tufts D. A least-squares algorithm for multipath time- delay estimation. IEEE Trans Signal Process 1994;42(11):3229–33. [http://dx.doi.](http://dx.doi.org/10.1109/78.330381) [org/10.1109/78.330381](http://dx.doi.org/10.1109/78.330381).
4. Benesty J. Adaptive eigenvalue decomposition algorithm for passive acoustic source localization. J Acoust Soc Am 2000;107(1):384–91. [http://dx.doi.org/10.](http://dx.doi.org/10.1121/1.428310)

[1121/1.428310](http://dx.doi.org/10.1121/1.428310).

= ⎪⎨ 1

2 1 2 1 2 1 2

1. [Haykin S. Radar array processing for angle of arrival estimation. In: Array signal](http://refhub.elsevier.com/S2590-0056(22)00075-3/sb5)

1{((*𝑡*1*, 𝑡*2 −1)*,* (*𝑡*1*, 𝑡*2)) ∈ *𝐸*}*𝐹* (*𝑡*1*, 𝑡*2 −1)

+1{((*𝑡*1 −1*, 𝑡*2)*,* (*𝑡*1*, 𝑡*2)) ∈ *𝐸*}*𝐹* (*𝑡*1 −1*, 𝑡*2)

⎪

⎪⎩+1{((*𝑡*1 −1*, 𝑡*2 −1)*,* (*𝑡*1*, 𝑡*2)) ∈ *𝐸*}*𝐹* (*𝑡*1 −1*, 𝑡*2 −1) (*𝑡*1*, 𝑡*2 *>* 0)*.*

[processing (A85-43960 21-32). Englewood Cliffs. 1985, p. 194–292.](http://refhub.elsevier.com/S2590-0056(22)00075-3/sb5)

1. Sakoe H, Chiba S. Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans Acoust Speech Signal Process 1978;26(1):43–9. <http://dx.doi.org/10.1109/TASSP.1978.1163055>.
2. Hale D. Dynamic warping of seismic images. Geophysics 2013;78(2):S105–15. <http://dx.doi.org/10.1190/geo2012-0327.1>.
3. [Mount DW. Bioinformatics - sequence and genome analysis. 2nd ed.. Cold Spring](http://refhub.elsevier.com/S2590-0056(22)00075-3/sb8) [Harbor Laboratory Press; 2004.](http://refhub.elsevier.com/S2590-0056(22)00075-3/sb8)
4. [Juang B-H. On the hidden Markov model and dynamic time warping for speech](http://refhub.elsevier.com/S2590-0056(22)00075-3/sb9) [recognition—A unified view. AT&T Bell Lab Tech J 1984;63(7):1213–43.](http://refhub.elsevier.com/S2590-0056(22)00075-3/sb9)
5. Hayashi T, Nakamura A. Propagation graph estimation from individuals’ time series of observed states. Sci Rep 2022;12(6078). [http://dx.doi.org/10.1038/](http://dx.doi.org/10.1038/s41598-022-10031-3) [s41598-022-10031-3](http://dx.doi.org/10.1038/s41598-022-10031-3).