Knowledge and Information Systems (2021) 63:2621–2644
https://doi.org/10.1007/s10115-021-01599-9

REGULAR PAPER

Bypassing combinatorial explosions in equivalence structure extraction

Seiya Satoh[1] · Hiroshi Yamakawa[2]

Received: 30 January 2021 / Revised: 13 July 2021 / Accepted: 17 July 2021 /
Published online: 13 August 2021

Abstract
Equivalence structure (ES) extraction enables us to determine correspondence relations within a dataset or between multiple datasets. Applications of ES extraction include the analysis of time series data, preprocessing of imitation learning, and preprocessing of transfer learning. Currently, pairwise incremental search (PIS) is the fastest method to extract ESs; however, a combinatorial explosion can occur when employing this method. In this paper, we show that combinatorial explosion is a problem that occurs in the PIS, and we propose a new method where this problem does not occur. We evaluate the proposed method via experiments; the results show that our proposed method is 39 times faster than the PIS for synthetic datasets where a 20-dimensional ES exists. For the experiment using video datasets, the proposed method enabled us to obtain a 29-dimensional ES, whereas the PIS did not because the memory usage reached its limit when the number of dimensions was 9. In this experiment, the total processing time for our proposed method up to 29 dimensions was 6.3 times shorter than that for PIS up to even 8 dimensions.

Keywords Time series · Equivalence structure · Combinatorial explosion · Motif discovery

1 Introduction

Equivalence structure (ES) extraction can help determine correspondence relations between two multidimensional sequences [9]. An ES is a set of tuples that indicates multidimensional sequences that can be considered equivalent. The applications of ES extraction include the analysis of time series data, preprocessing of imitation learning [3,12,14,17], and preprocessing of transfer learning [7,13,15,19]. As an example of the preprocessing of imitation learning, if the correspondence relation between the dimensions of the student and those of the teacher is known, imitation learning can be performed more efficiently. As preprocessing of transfer learning, it is useful to determine a correspondence relation between the dimen-

✉ Seiya Satoh
seiya.satoh@mail.dendai.ac.jp

1 School of Science and Engineering, Tokyo Denki University, Hiki-gun, Saitama, Japan

2 Department of Technology Management for Innovation, The University of Tokyo, Bunkyo-ku, Tokyo, Japan

sions of the source domain and those of the target domain. In transfer learning, when the representation of the dimensions is compressed by a deep neural network etc., the representation can be entangled (owing to the redundancy of a deep neural network), and therefore, correspondence relations can be found more effectively by using a disentanglement method [2,5,6]. Disentanglement technology has been being studied for time series data [16].

Motif discovery is a technique similar to ES extraction [1,4,18]. Although some motif discovery techniques can extract similar subsequences from two multidimensional sequences, the correspondence between each dimension of the two multidimensional sequences must be known. The ES extraction determines the correspondence between each dimension itself.

The existing methods for ES extraction include the brute-force search (BFS), incremental search (IS) [8,10], and pairwise incremental search (PIS) [9] exist. Because a combinatorial explosion of the number of comparisons occurs in BFS, IS was proposed to obtain K-dimensional ESs based on $(K - 1)$-dimensional ESs and suppress the combinatorial explosion. However, an ES obtained by IS may be a subset of another ES, and it takes a long time to remove such a subset. Therefore, PIS was proposed where ESs are decomposed into pairs called equivalent pairs (EPs), which are obtained instead of ESs. Although PIS is considerably faster than IS, the combinatorial explosion remains a problem when the number of dimensions of EPs is high.

In this paper, we show that a combinatorial explosion remains a problem even in the procedure of PIS, and we propose a new method where the combinatorial explosion does not occur. In particular, a large number of unnecessary EPs are found in the procedure of PIS, and it causes a combinatorial explosion; therefore, we propose a new method where the problem does not occur. In addition, we evaluate our proposed method by conducting three experiments.

This paper is organized as follows. In Sect. 2, we describe ES extraction. In Sect. 3, we describe the existing methods. In Sect. 4, we discuss the problem in the procedure of PIS, and we present the proposed method. Then, we evaluate the proposed method via three experiments in Sect. 5. Finally, we offer conclusions and discuss future works in Sect. 6.

2 Equivalence structure extraction

The ES extraction enables us to determine correspondence relations within a dataset or between multiple datasets [9]. In this paper, we consider only ES extraction that determines corresponding relations between two datasets. Therefore, the input of ES extraction is two datasets and the output is a set of ESs.

A K-dimensional ES is a set of K-tuples that indicate K-dimensional sequences. Here, we consider only the case where K is an integer greater than one. The elements of a K-tuple are K IDs for K sequences. A K-dimensional ES indicates K-dimensional sequences that are *equivalent*. The standard for *equivalence* is determined based on the subsequences of the K-dimensional sequences. Figure 1 shows an example of a three-dimensional ES.

In the example, a three-dimensional sequence specified by three-tuple $\langle$A1, A2, A3$\rangle$ and a three-dimensional sequence specified by three-tuple $\langle$B4, B2, B1$\rangle$ are considered equivalent because one subsequence of the former sequence is very similar to one subsequence of the latter one. Further, one subsequence each of sequences A1, A2, and A3 are very similar to one subsequence each of B1, B2, and B4, respectively. However, none of the subsequences of the three-dimensional sequence specified by $\langle$A1, A2, A3$\rangle$, and the three-dimensional sequence specified by $\langle$B1, B2, B4$\rangle$ are similar, and the two three-dimensional sequences

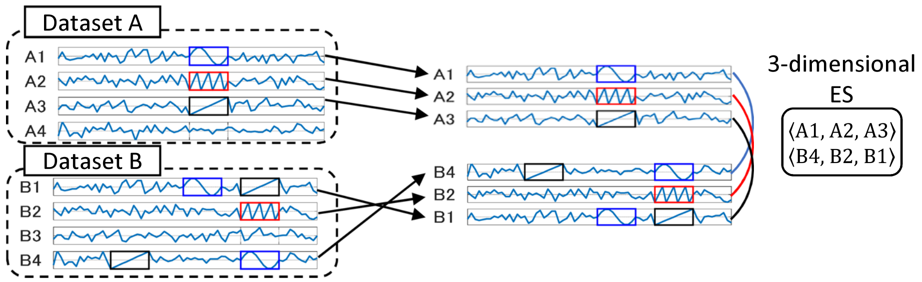


Fig. 1 Illustration of ES extraction

are not considered equivalent. Thus, comparisons between one-dimensional sequences are not sufficient, and that between subsequences of K-dimensional sequences is required when determining the K-dimensional ES.

2.1 A standard for *equivalence*

The ES extraction requires a standard that multidimensional sequences are *equivalent*. In this study, the standard for *equivalence* described below is used; however, it may be better to use a different standard depending on the application.

First, we consider a dataset $\{\boldsymbol{x}_{\mathrm{A}1}, ..., \boldsymbol{x}_{\mathrm{A}N_1}\}$ that has N_1 sequences and the lengths are all T_1, and a dataset $\{\boldsymbol{x}_{\mathrm{B}1}, ..., \boldsymbol{x}_{\mathrm{B}N_2}\}$ that has N_2 sequences and the lengths are all T_2. Here, we call the two datasets as dataset A and dataset B. A subsequence is described as

$$\begin{aligned} \boldsymbol{z}_k^{(t)} &\equiv (z_k^{(t,1)}, ..., z_k^{(t,\tau)})^{\mathrm{tr}} \\ &= (x_k^{(t)}, ..., x_k^{(t+\tau-1)})^{\mathrm{tr}} - \frac{1}{\tau}\sum_{t'=1}^{\tau} x_k^{(t+t'-1)} \end{aligned} \tag{1}$$

where k denotes an ID for a sequence, $t \in \{1, ..., T_1 - \tau + 1\}$ if $k \in \{\mathrm{A}1, ..., \mathrm{A}N_1\}$, $t \in \{1, ..., T_2 - \tau + 1\}$ if $k \in \{\mathrm{B}1, ..., \mathrm{B}N_2\}$, τ denotes the length of a subsequence, the symbol $\equiv$ denotes *is equal by definition to*, and $\boldsymbol{a}^{\mathrm{tr}}$ denotes the transposition of a vector $\boldsymbol{a}$.

We use the similarity function

$$s(\boldsymbol{v}_1, \boldsymbol{v}_2) = \frac{1}{\beta}\left(\sum_{t=1}^{T_1-\tau+1} w_{\boldsymbol{v}_1}^{(t)} h_{\boldsymbol{v}_1,\boldsymbol{v}_2}^{(t)} + \sum_{t=1}^{T_2-\tau+1} w_{\boldsymbol{v}_2}^{(t)} h_{\boldsymbol{v}_2,\boldsymbol{v}_1}^{(t)}\right) \tag{2}$$

where

$$\beta \equiv \sum_{t=1}^{T_1-\tau+1} w_{\boldsymbol{v}_1}^{(t)} + \sum_{t=1}^{T_2-\tau+1} w_{\boldsymbol{v}_2}^{(t)}, \tag{3}$$

$$w_{\boldsymbol{v}_i}^{(t)} \equiv \frac{1}{\tau}\sum_{t'=1}^{\tau}\sqrt{\sum_{k=1}^{K}\left\{z_{v_{i,k}}^{(t,t')}\right\}^2}, \tag{4}$$

$$h_{\boldsymbol{v}_i,\boldsymbol{v}_{i'}}^{(t)} \equiv h(\theta_{\mathrm{MSV}} - \mathrm{MSV}_{\boldsymbol{v}_i,\boldsymbol{v}_{i'}}^{(t)}), \tag{5}$$

$$\mathrm{MSV}_{\boldsymbol{v}_i,\boldsymbol{v}_{i'}}^{(t)} \equiv \min(\mathrm{MSV}_{\boldsymbol{v}_i,\boldsymbol{v}_{i'}}^{(t,t')} | t' \in \{1, ..., T_{i'} - \tau + 1\}), \tag{6}$$

$$\mathrm{MSV}^{(t,t')}_{\boldsymbol{v}_i,\boldsymbol{v}_{i'}} \equiv \frac{1}{\tau K}\sum_{k=1}^{K}\left|z^{(t)}_{v_{i,k}} - z^{(t')}_{v_{i',k}}\right|^2, \tag{7}$$

$i \in \{1, 2\}, i' \in \{1, 2\} \setminus \{i\}$, and $\boldsymbol{v}_1 \equiv \langle v_{1,1}, ..., v_{1,K}\rangle$ specifies a K-dimensional sequence composed of K sequences in dataset A, and $\boldsymbol{v}_2 \equiv \langle v_{2,1}, ..., v_{2,K}\rangle$ specifies a K-dimensional sequence composed of K sequences in dataset B. $\mathrm{MSV}^{(t,t')}_{\boldsymbol{v}_i,\boldsymbol{v}_{i'}}$ in Eq. (7) is the mean-square value (MSV); $\mathrm{MSV}^{(t)}_{\boldsymbol{v}_i,\boldsymbol{v}_{i'}}$ in Eq. (6) is the minimum MSV when comparing the subsequence of the K-dimensional sequence specified by $\boldsymbol{v}_i$ starting at t with all subsequences of the K-dimensional sequence specified by $\boldsymbol{v}_{i'}$. The value of h in Eq. (5) is the Heaviside step function, and $h^{(t)}_{\boldsymbol{v}_1,\boldsymbol{v}_2}$ in Eq. (5) is 1 if $\mathrm{MSV}^{(t)}_{\boldsymbol{v}_1,\boldsymbol{v}_2} < \theta_{\mathrm{MSV}}$ and 0 otherwise. Further, θ_{MSV} is a threshold that should be adjusted depending on the application. $w^{(t)}_{\boldsymbol{v}_1}$ in Eq. (4) indicates the weight of the subsequence of the K-dimensional sequence specified by v_1; the larger the variance of the subsequence, the larger is the value. When $h^{(t)}_{\boldsymbol{v}_1,\boldsymbol{v}_2}$ is 1 in a place where the weight is large, it is more likely to be regarded as *equivalent*. The value $s(\boldsymbol{v}_1, \boldsymbol{v}_2)$ is in the interval [0, 1]. Here, the sequences specified by $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are regarded as equivalent if the value of $s(\boldsymbol{v}_1, \boldsymbol{v}_2)$ is greater than the threshold θ_s. In this standard, it is necessary to set the length τ of a subsequence, the thresholds θ_{MSV} and θ_s. Here, the MSV is used to compare the subsequences as in Eq. (7); however, it may be better to use a different calculation such as dynamic time warping [1,11]. Note that we define the similarity function $s(\boldsymbol{v}_1^{(K)}, \boldsymbol{v}_2^{(K)})$ instead of the dissimilarity function $d(\boldsymbol{v}_1^{(K)}, \boldsymbol{v}_2^{(K)})$ unlike the previous study [9], but the functionality is the same because $d(\boldsymbol{v}_1^{(K)}, \boldsymbol{v}_2^{(K)}) = 1 - s(\boldsymbol{v}_1^{(K)}, \boldsymbol{v}_2^{(K)})$.

2.1.1 Properties of the similarity function

Even if $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ in Eq. (2) are transposed, the value of $s(\boldsymbol{v}_1, \boldsymbol{v}_2)$ is the same; however, for simplicity, we consider only the case where the tuple of the first argument of s is a tuple to specify a multidimensional sequence of dataset A, and the tuple of the second argument is used to specify that of dataset B. Then, the similarity function $s(\boldsymbol{v}_1, \boldsymbol{v}_2)$ has the symmetries

$$s(\boldsymbol{v}_1^{(K)}, \boldsymbol{v}_2^{(K)}) = s\left(\left\langle v_{1,p_1}, ..., v_{1,p_K}\right\rangle, \left\langle v_{2,p_1}, ..., v_{2,p_K}\right\rangle\right) \tag{8}$$

where

$$\boldsymbol{v}_1^{(K)} = \langle v_{1,1}, ..., v_{1,K}\rangle, \quad \boldsymbol{v}_2^{(K)} = \langle v_{2,1}, ..., v_{2,K}\rangle, \quad \langle p_1, ..., p_K\rangle \in P^{(K)}, \tag{9}$$

and $P^{(K)}$ denotes the set of all K-permutations of $\{1, ..., K\}$. Therefore, if the elements of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are replaced similarly and simultaneously, the value of s will be the same.

3 Existing methods

The BFS, IS [8,10], and PIS [9] are the existing methods for ES extraction.

3.1 Brute-force search

In the BFS, all K-dimensional sequences are compared to obtain K-dimensional ESs. When searching K-dimensional ESs between two datasets, the number of calculations of the similarity function is ${}_{N_\mathrm{A}}\mathrm{P}_K\, {}_{N_\mathrm{B}}\mathrm{P}_K$, where the number of sequences for dataset A is N_A, that for dataset B is N_B, and ${}_N\mathrm{P}_K$ is the number of K-permutations of $\{1, ..., N\}$. Considering the similarity properties described in Sect. 2.1.1, the number of calculations of the similarity function can be reduced to $\frac{{}_{N_\mathrm{A}}\mathrm{P}_{K\,N_\mathrm{B}}\mathrm{P}_\mathrm{K}}{K!}$ $(= {}_{N_\mathrm{A}}\mathrm{C}_{K\,N_\mathrm{B}}\mathrm{P}_K = {}_{N_\mathrm{A}}\mathrm{P}_{K\,N_\mathrm{B}}\mathrm{C}_K)$ where ${}_N\mathrm{C}_K$ is the number of K-combinations of $\{1, ..., N\}$. However, even if the number of the calculations is reduced, the combinatorial explosion remains a problem.

3.2 Incremental search

The IS was proposed because the combinatorial explosion of the number of calculations of the similarity function occurs in the BFS [10]. In IS, K-dimensional ESs are searched from the candidates generated using $(K-1)$-dimensional ESs. Figure 2 shows a conceptual diagram of IS. Algorithm 1 shows the procedure of IS. We omit the details of how to generate candidates in Step 5; these details (method and validity) are provided in [8]. An ES obtained by IS can be a subset of another ES, and such a subset needs to be removed. Therefore, it requires a considerable amount of time to determine whether an ES is a subset of another ES and to then delete it. Figure 3 shows an example where such subsets exist. In the figure, ES 1 is an ES with four tuples, and ES 2 is a subset of ES 1 and must be deleted. ES 3 is the same as ES 2 when the first and third elements of all tuples in ES 3 are transposed. Therefore, ES 1 and ES 3 may appear to be different at first glance; however, ES 3 needs to be removed.

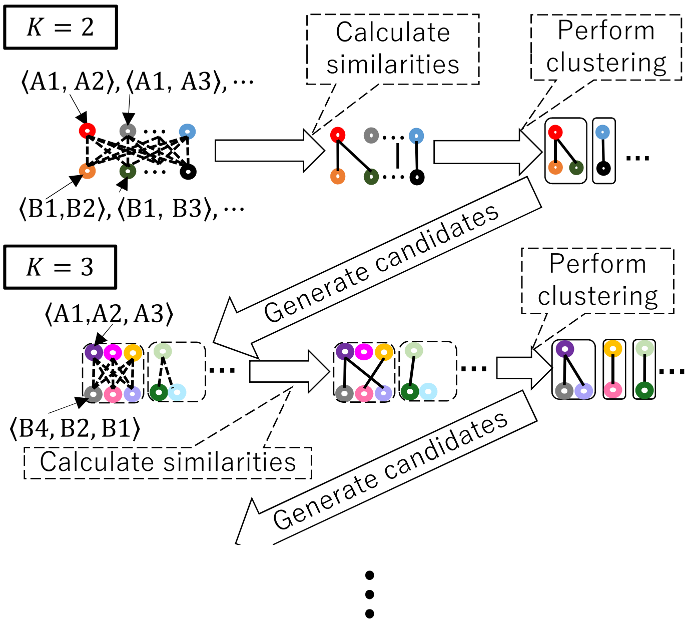


Fig. 2 Conceptual diagram of IS. A circle, rounded square, and dotted rounded square represent a tuple, an ES, and a candidate for an ES, respectively

Algorithm 1 Incremental search (IS)

```
1: $K \leftarrow 2$
2: Calculate all similarities between two-dimensional sequences and obtain two-dimensional ESs (the number
   of calculations can be reduced considering the properties of similarity function $s$ described in Sect. 2.1.1)
3: while $K$-dimensional ESs exist do
4:     $K \leftarrow K + 1$
5:     Generate candidates for $K$-dimensional ESs using $(K - 1)$- and two-dimensional ESs
6:     Calculate similarities between $K$-dimensional sequences specified by the tuples in each candidate
7:     Obtain $K$-dimensional ESs based on the calculated similarities
8: end while
```

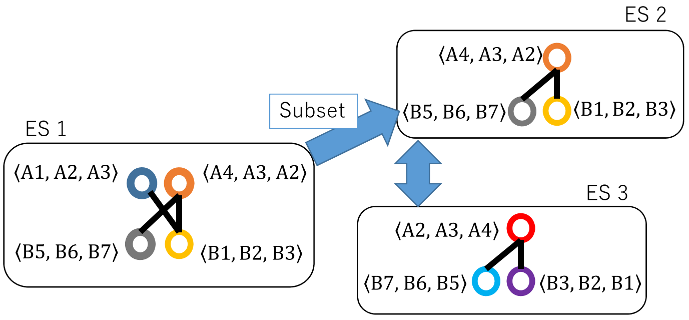


Fig. 3 Example of subsets of ESs

3.3 Pairwise incremental search

The PIS was proposed to overcome the problem in IS: ESs can be subsets of other ESs. In PIS, an ES is split into pairs [9]. Figure 4 shows a conceptual diagram of PIS. A pair obtained by decomposing an ES is called an *equivalent pair (EP)*. For example, ES {⟨A1, A2⟩, ⟨B1, B2⟩, ⟨B4, B3⟩} is decomposed into two EPs {⟨A1, A2⟩, ⟨B1, B2⟩}, and {⟨A1, A2⟩, ⟨B1, B2⟩}. Since an EP is a pair, it can be represented as a two-tuple whose elements are tuples; however, for convenience, it is represented as a set here. In the procedure of the PIS, the problem of subsets of ESs does not occur because the number of elements of an EP is always two. Further, ESs can be built based on EPs if necessary.

Algorithm 2 shows the procedure of PIS. Here, the processes in Steps 2, 5, 6, and 7 can be calculated in parallel, and therefore, they were processed in parallel in the following experiments.

The simplest method to generate candidates in Step 5 is to connect a connectable ID to a tuple in a $(K - 1)$-dimensional ES. For example, IDs connectable to tuple ⟨A1, A2⟩ are

Algorithm 2 Pairwise incremental search (PIS)

```
1: $K \leftarrow 2$
2: Calculate all similarities between two-dimensional sequences and obtain two-dimensional EPs (the number
   of calculations can be reduced considering the properties of similarity function $s$ described in Sect. 2.1.1)
3: while there exist $K$-dimensional EPs do
4:     $K \leftarrow K + 1$
5:     Generate candidates for $K$-dimensional EPs using $(K - 1)$- and two-dimensional EPs
6:     Calculate similarities between $K$-dimensional sequences specified by the tuples in each candidate
7:     Obtain $K$-dimensional EPs based on the calculated similarities
8: end while
```

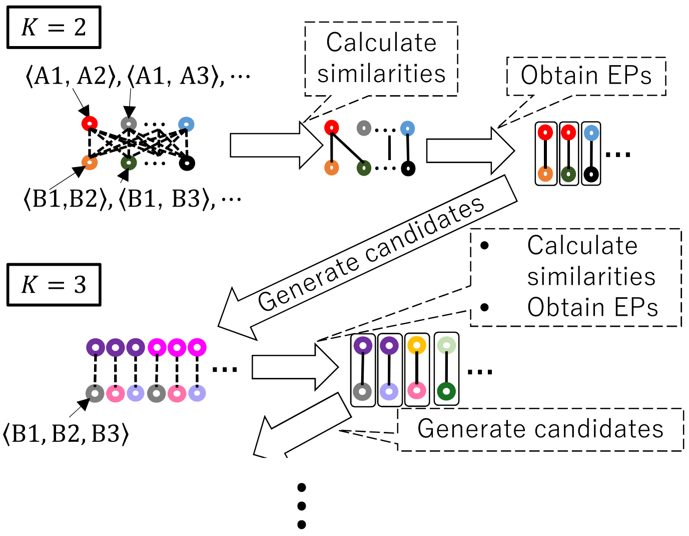


Fig. 4 Conceptual diagram of pairwise incremental search. A circle and rounded square represent a tuple and an ES, respectively

A3, ..., AN_A, and we consider ⟨A1, A2, A3⟩, ..., ⟨A1, A2, AN_A⟩. However, we use a more efficient approach to generate candidates, as described below.

3.3.1 Candidate generation

A candidate $\widetilde{E}^{(K)}_{r,\boldsymbol{r}}$ for a K-dimensional EP is generated using

$$\begin{aligned}\widetilde{E}^{(K)}_{r,\boldsymbol{r}} &= g'(E^{(K-1)}_r, \boldsymbol{r})\\ &= \left\{\langle v^*_1, ..., v^*_{K-1}, i\rangle \middle| \bigwedge_{k\in\{1,...,K-1\}} v^*_k \neq i, \bigwedge_{k\in\{1,...,K-1\}} \langle v^*_k, i\rangle \in E^{(2)}_{r_k},\right.\\ &\qquad \left.\langle v^*_1, ..., v^*_{K-1}\rangle \in E^{(K-1)}_r\right\} \end{aligned} \tag{10}$$

where $E^{(K-1)}_r$ denotes a $(K-1)$-dimensional EP, $\boldsymbol{r} \in \{\langle r_1, ..., r_{K-1}\rangle \mid r_1, ..., r_{K-1} \in \{1, ..., R^{*(2)}\}\}$. For example, when searching for three-dimensional EPs, if $E^{(2)}_1 = \{\langle A1, A2\rangle, \langle B4, B5\rangle\}$, $E^{(2)}_2 = \{\langle A1, A3\rangle, \langle B4, B6\rangle\}$, $E^{(2)}_3 = \{\langle A2, A3\rangle, \langle B5, B6\rangle\}$, $E^{(2)}_4 = \{\langle A1, A2\rangle, \langle B5, B6\rangle\}$ exist, $r = 1$, and $\boldsymbol{r} = \langle 2, 3\rangle$, Then, $\widetilde{E}^{(K)}_{r,\boldsymbol{r}} = \{\langle A1, A2, A3\rangle, \langle B4, B5, B6\rangle\}$. In addition, if $r = 1, \boldsymbol{r} = \langle 4, 3\rangle$, then $\widetilde{E}^{(K)}_{r,\boldsymbol{r}} = \{\langle A1, A2, A3\rangle\}$. The maximum number of elements of $\widetilde{E}^{(K)}_{r,\boldsymbol{r}}$ is 2, and when the number is 1 or 0, it is indeed excluded from the candidates.

It has been proven that candidate generation enables us to obtain all EPs that are the same as all EPs obtained by BFS under the following assumption [9]:

Assumption 1

$$\forall k \in \{1, ..., K\} : \forall r \in \{1, ..., R^{*(K)}\} :$$

$$\exists q_{\{k\}} \in \{1, ..., R^{*(K-1)}\} : \quad \mathrm{drop}(E_r^{(K)}, k) = E_{q_{\{k\}}}^{(K-1)} \tag{11}$$

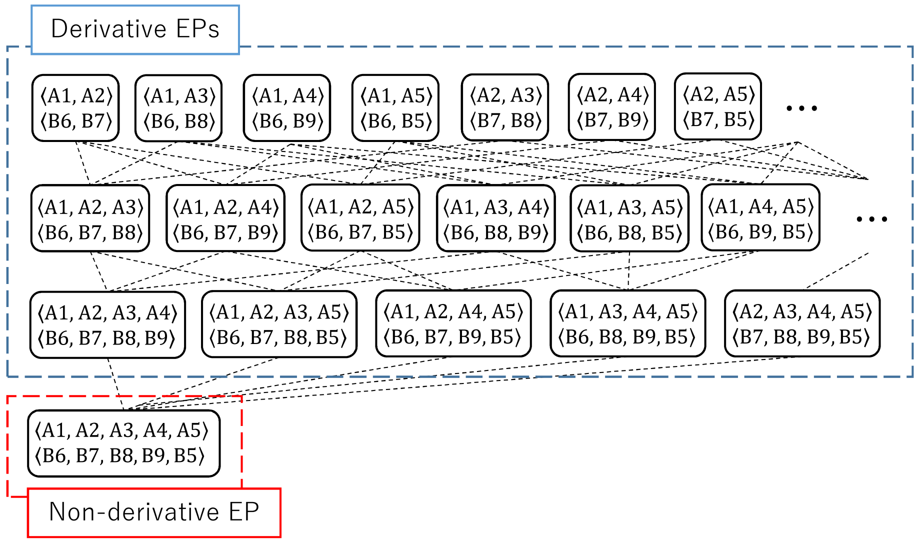
where $E_1^{(K)}, ..., e_{R^{*(K)}}^{(K)}$ are the K-dimensional EPs, and $E_1^{(K-1)}, ..., E_{R^{*(K-1)}}^{(K-1)}$ are the $(K-1)$-dimensional EPs.

For example, when $\{\langle A1, A2, A3\rangle, \langle B4, B5, B6\rangle\}$ is an EP, $\{\langle A1, A2\rangle, \langle B4, B5\rangle\}$, $\{\langle A1, A3\rangle, \langle B4, B6\rangle\}$, $\{\langle A2, A3\rangle, \langle B5, B6\rangle\}$ are all EPs under the assumption. The assumption is considered to hold to a certain extent, but the extent to which it holds depends on the standard for equivalence, noises in data, and so on.

4 A problem with PIS and our proposed method

4.1 Derivative EPs

Although PIS is faster than IS, derivative EP (DEP) exists and a combinatorial explosion can occur. If a K-dimensional EP exists and Assumption 1 holds, then $\{\langle v_{c_1}, ..., v_{c_k}\rangle \mid \boldsymbol{v}^{(K)} \in E^{*(K)}_{c'}\}$ are all EPs where $\langle c_1, ..., c_k\rangle \in C_k^{(K)}$, $C_k^{(K)}$ denotes the set of all k combinations of $\{1, ..., K\}$, and $k \in \{2, ..., K-1\}$. Here, EPs that can be expressed in this manner are called *derivative EPs (DEPs)*, and the other EPs are called *non-derivative EPs (NDEPs)*. This means that if Assumption 1 holds and a K-dimensional NDEP exists, then, the number of its DEPs is $\sum_{k=2}^{K-1} {}_K\mathrm{C}_k$ where ${}_K\mathrm{C}_k$ is the number of k-combinations of $\{1, ..., K\}$. Therefore, the combinatorial explosion occurs when there exist high-dimensional EPs. Figure 5 shows an example where an NDEP and its DEPs exist. In the figure, there exits NDEP $\{\langle A1, A2, A3, A4, A5\rangle, \langle B6, B7, B8, B9, B10\rangle\}$ and its DEPs such as $\{\langle A1, A2, A3, A4\rangle, \langle B6, B7, B8, B9\rangle\}$ and $\{\langle A2, A3, A5\rangle, \langle B7, B8, B10\rangle\}$. In this case, the number of DEPs is $\sum_{k=2}^{K-1} {}_K\mathrm{C}_k$ where $K = 5$, and the number of calculations of similarities is the same number as that of



Derivative EPs
⟨A1, A2⟩ ⟨B6, B7⟩
⟨A1, A3⟩ ⟨B6, B8⟩
⟨A1, A4⟩ ⟨B6, B9⟩
⟨A1, A5⟩ ⟨B6, B5⟩
⟨A2, A3⟩ ⟨B7, B8⟩
⟨A2, A4⟩ ⟨B7, B9⟩
⟨A2, A5⟩ ⟨B7, B5⟩
...
⟨A1, A2, A3⟩ ⟨B6, B7, B8⟩
⟨A1, A2, A4⟩ ⟨B6, B7, B9⟩
⟨A1, A2, A5⟩ ⟨B6, B7, B5⟩
⟨A1, A3, A4⟩ ⟨B6, B8, B9⟩
⟨A1, A3, A5⟩ ⟨B6, B8, B5⟩
⟨A1, A4, A5⟩ ⟨B6, B9, B5⟩
...
⟨A1, A2, A3, A4⟩ ⟨B6, B7, B8, B9⟩
⟨A1, A2, A3, A5⟩ ⟨B6, B7, B8, B5⟩
⟨A1, A2, A4, A5⟩ ⟨B6, B7, B9, B5⟩
⟨A1, A3, A4, A5⟩ ⟨B6, B8, B9, B5⟩
⟨A2, A3, A4, A5⟩ ⟨B7, B8, B9, B5⟩
⟨A1, A2, A3, A4, A5⟩ ⟨B6, B7, B8, B9, B5⟩
Non-derivative EP

Fig. 5 Non-derivative EP and its derivative EPs

Algorithm 3 Pairwise dimension-first search (PDFS)

```
1: Obtain two-dimensional EPs in the same way as PIS
2: Generate candidates for three-dimensional EPs
3: while There exist candidates do
4:     Select $n_{para}$ nonderivative candidates in the descending order of the number of dimensions and delete derivative candidates
       (Select all nonderivative candidates if the number of nonderivative candidates is less than $n_{para}$)
5:     Calculate similarities between $K$-dimensional sequences specified by the tuples in each selected candidate and obtain EPs based on the similarities
6:     if EPs are obtained then
7:         Generate candidates based on the obtained EPs
8:     end if
9: end while
```

DEPs. Therefore, the number of calculations of similarities rapidly increases along with the increase in the number K of dimensions of an NDEP.

4.2 Pairwise dimension-first search

As described in Sect. 4.1, the combinatorial explosion of the number of DEPs occurs in the procedure of PIS when high-dimensional EPs exist. Therefore, we propose a search method where a combinatorial explosion does not occur.

In our proposed method, two-dimensional EPs are obtained in the same manner as that in PIS, and candidates for three-dimensional EPs are generated. Next, in PIS, all three-dimensional EPs are obtained; however, in our proposed method, not all EPs are obtained and candidates for four-dimensional EPs are generated; four-dimensional EPs are obtained before obtaining the other three-dimensional EPs. Then, an EP with a large number of dimensions is preferentially searched. Thus, we call our proposed method pairwise dimension-first search (PDFS). Figure 6 shows the processing flows of PIS and PDFS.

Algorithm 3 shows the algorithm of PDFS. For the candidate generation in Step 7, we can use function g provided in Eq. (10) as done in PIS. In addition, the processes in Steps 1, 2, 5, and 7 can be calculated in parallel, and therefore, they were processed in parallel in the following experiments. If the n_{para} in Step 4 is 1, the number of calculations of the similarity function should be the smallest. However, the total processing time can be faster when the n_{para} is greater than 1 because Steps 5 and 7 can be processed in parallel. In Step 4, the determination of whether a candidate is *derivative* is based on the EPs obtained at that time.

In the example shown in Fig. 6b, all two-dimensional EPs are obtained, and all candidates for three-dimensional EPs are generated in Steps 1 and 2 of PDFS in the same way as PIS. In Step 4 of PDFS, $\{\langle A1, A2, A3\rangle, \langle B6, B7, B8\rangle\}$ is selected as a nonderivative candidate. Then, the similarity $s(\langle A1, A2, A3\rangle, \langle B6, B7, B8\rangle)$ is calculated, and EP $\{\langle A1, A2, A3\rangle, \langle B6, B7, B8\rangle\}$ is obtained in Step 5. In Step 7, candidates $\{\langle A1, A2, A3, A4\rangle, \langle B6, B7, B8, B9\rangle\}$ and $\{\langle A1, A2, A3, A5\rangle, \langle B6, B7, B8, B5\rangle\}$ are generated using the EP $\{\langle A1, A2, A3\rangle, \langle B6, B7, B8\rangle\}$. Go back to Step 4, $\{\langle A1, A2, A3, A4\rangle, \langle B6, B7, B8, B9\rangle\}$ is selected as a nonderivative candidate. Then, the similarity $s(\langle A1, A2, A3, A4\rangle, \langle B6, B7, B8, B9\rangle)$ is calculated, and EP $\{\langle A1, A2, A3, A4\rangle, \langle B6, B7, B8, B9\rangle\}$ is obtained in Step 5. In Step 7, $\{\langle A1, A2, A3, A4, A5\rangle, \langle B6, B7, B8, B9, B5\rangle\}$ is generated as a candidate using the EP $\{\langle A1, A2, A3, A4\rangle, \langle B6, B7, B8, B9\rangle\}$. Go back to Step 4, $\{\langle A1, A2, A3, A4, A5\rangle, \langle B6, B7, B8, B9, B5\rangle\}$ is selected as a nonderivative candidate. Then, the similarity $s(\langle A1, A2, A3, A4, A5\rangle, \langle B6, B7,$

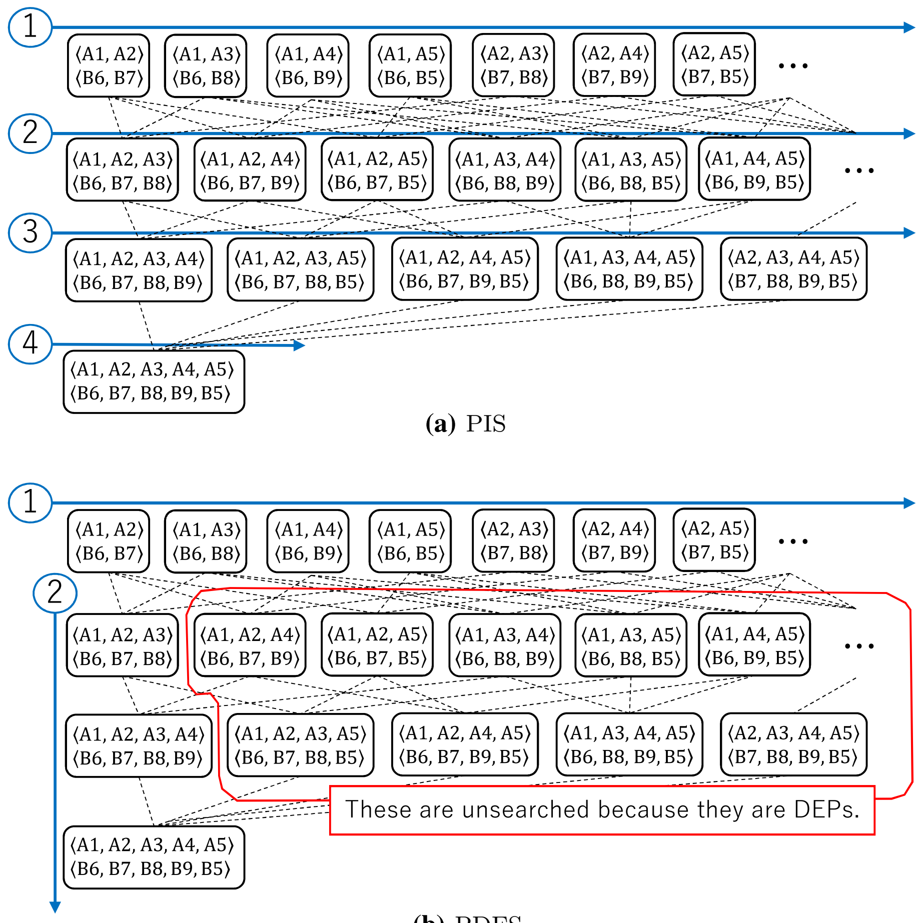

1
⟨A1, A2⟩ ⟨B6, B7⟩
⟨A1, A3⟩ ⟨B6, B8⟩
⟨A1, A4⟩ ⟨B6, B9⟩
⟨A1, A5⟩ ⟨B6, B5⟩
⟨A2, A3⟩ ⟨B7, B8⟩
⟨A2, A4⟩ ⟨B7, B9⟩
⟨A2, A5⟩ ⟨B7, B5⟩
…
2
⟨A1, A2, A3⟩ ⟨B6, B7, B8⟩
⟨A1, A2, A4⟩ ⟨B6, B7, B9⟩
⟨A1, A2, A5⟩ ⟨B6, B7, B5⟩
⟨A1, A3, A4⟩ ⟨B6, B8, B9⟩
⟨A1, A3, A5⟩ ⟨B6, B8, B5⟩
⟨A1, A4, A5⟩ ⟨B6, B9, B5⟩
…
3
⟨A1, A2, A3, A4⟩ ⟨B6, B7, B8, B9⟩
⟨A1, A2, A3, A5⟩ ⟨B6, B7, B8, B5⟩
⟨A1, A2, A4, A5⟩ ⟨B6, B7, B9, B5⟩
⟨A1, A3, A4, A5⟩ ⟨B6, B8, B9, B5⟩
⟨A2, A3, A4, A5⟩ ⟨B7, B8, B9, B5⟩
4
⟨A1, A2, A3, A4, A5⟩ ⟨B6, B7, B8, B9, B5⟩
(a) PIS
These are unsearched because they are DEPs.
(b) PDFS

Fig. 6 Processing flows of PIS and PDFS where an NDEP exists and $n_{para} = 1$

B8, B9, B5⟩) is calculated, and EP {⟨A1, A2, A3, A4, A5⟩, ⟨B6, B7, B8, B9, B5⟩} is obtained in Step 5. In Step 7, no candidate is generated using the EP {⟨A1, A2, A3, A4, A5⟩, ⟨B6, B7, B8, B9, B5⟩}. Go back to Step 4, nonderivative candidates are searched from candidates such as {⟨A1, A2, A4⟩, ⟨B6, B7, B9⟩} and {⟨A1, A2, A5⟩, ⟨B6, B7, B5⟩}, but none of the candidates are nonderivative candidates. Therefore, the similarities of the candidates are not calculated. In addition, sets {⟨A1, A2, A4, A5⟩, ⟨B6, B7, B9, B5⟩}, {⟨A1, A3, A4, A5⟩, ⟨B6, B8, B9, B5⟩}, and {⟨A2, A3, A4, A5⟩, ⟨B7, B8, B9, B5⟩} shown in Fig. 6b do not become even candidates and the calculations of the similarities can be skipped, while the similarities of all pairs shown in Fig. 6a are calculated in the procedure of PIS.

4.3 Number of calculations of similarity function in PIS and PDFS

First, we consider the case where Assumption 1 holds and a K-dimensional NDEP exists; we refer to this as case 1. Then, the number of DEPs obtained by PIS is $\sum_{k=2}^{K-1} {}_K C_k$. In addition, because all two-dimensional sequences are compared, the number of calculations of the similarity function in PIS is

$$N_{\mathrm{A}}(N_{\mathrm{A}}-1)N_{\mathrm{B}}(N_{\mathrm{B}}-1)/2+\sum_{k=3}^{K}{}_{K}\mathrm{C}_{k} \tag{12}$$

where the number of sequences for dataset A is N_{A} and that for dataset B is N_{B}. The first term of Eq. (12) is the number of calculations of similarities when obtaining two-dimensional EPs. The number of calculations of similarities when obtaining k-dimensional EPs is ${}_{K}\mathrm{C}_{k}$ where $k \in \{3, ..., K-1\}$. Because the number of K-dimensional EPs is one, which is an NDEP, the number of calculations of similarities when obtaining K-dimensional EPs is one, which is ${}_{K}\mathrm{C}_{K}$. Therefore, the total number of calculations of similarities for PIS is Eq. (12).

When n_{para} in Step 4 of Algorithm 3 is 1, the number of calculations of the similarity function in PDFS in case 1 is

$$N_{\mathrm{A}}(N_{\mathrm{A}}-1)N_{\mathrm{B}}(N_{\mathrm{B}}-1)/2+K-2 \tag{13}$$

as seen in Fig. 6. As is the case with PIS, the first term of Eq. (13) is the number of calculations of similarities when obtaining two-dimensional EPs. In the procedure of PDFS, not all three-dimensional EPs are obtained, but four-dimensional EPs are searched right after obtaining a three-dimensional EP. Therefore, the number of calculations of similarities when obtaining three-dimensional EPs is one. Similarly, the number of calculations of similarities when obtaining k-dimensional EPs is one where $k \in \{3, ..., K\}$. Consequently, the total number of calculations of similarities for PDFS is Eq. (13).

Next, we consider the case where Assumption 1 holds, the maximum number of dimensions of NDEPs is $K_{\max}$, the number of each K-dimensional NDEPs is M_K, and an ID for a sequence does not appear in NDEPs or appears only once. Here, we refer to this as case 2. The number of calculations of the similarity function in PIS is then

$$N_{\mathrm{A}}(N_{\mathrm{A}}-1)N_{\mathrm{B}}(N_{\mathrm{B}}-1)/2+\sum_{K=3}^{K_{\max}}M_K\left(\sum_{k=3}^{K}{}_{K}\mathrm{C}_{k}\right). \tag{14}$$

The number of calculations of the similarity function in PDFS is

$$N_{\mathrm{A}}(N_{\mathrm{A}}-1)N_{\mathrm{B}}(N_{\mathrm{B}}-1)/2+\sum_{K=3}^{K_{\max}}M_K(K-2). \tag{15}$$

Thus, the combinatorial explosion occurs in the PIS even when only one K-dimensional NDEP exists and K is large; however, it does not occur in the procedure of PDFS even if multiple NDEPs exist. Tables 1 and 2 summarize the numbers of calculations of the similarity function in BFS, PIS, and PDFS for cases 1 and 2.

Table 1 Number of calculations of the similarity function in case 1, where a K-dimensional EP exits

BFS	${}_{N_{\mathrm{A}}}\mathrm{P}_{K}\,{}_{N_{\mathrm{B}}}\mathrm{P}_{K}/K!$[1]
PIS	$N_{\mathrm{A}}(N_A-1)N_{\mathrm{B}}(N_{\mathrm{B}}-1)/2+\sum_{k=3}^{K}{}_{K}\mathrm{C}_{k}$
PDFS	$N_{\mathrm{A}}(N_{\mathrm{A}}-1)N_{\mathrm{B}}(N_{\mathrm{B}}-1)/2+K-2$

[1]Note that the actual number can be more than this number because K is usually unknown

Table 2 Number of calculations of the similarity function in case 2, where M_K K-dimensional EPs exist

BFS	$\sum_{K=2}^{K_{\max}} {}_{N_A}P_K \, {}_{N_B}P_K / K!$
PIS	$N_A(N_A-1)N_B(N_B-1)/2 + \sum_{K=3}^{K_{\max}} M_K \left(\sum_{k=3}^{K} {}_K C_k\right)$
PDFS	$N_A(N_A-1)N_B(N_B-1)/2 + \sum_{K=3}^{K_{\max}} M_K(K-2)$

5 Experiments

We conducted three experiments to evaluate PDFS. For the experiments, we used a system with an Intel(R) Core(TM) i9-9900X CPU (10 cores) @ 3.50 GHz and 32.0 GB RAM; we used MATLAB R2019b as the programming language. Further, we used Parallel Computing Toolbox version 7.1 in the MATLAB R2019b to perform parallel computing. The optimum number of n_{para} in Step 4 of Algorithm 3 may differ depending on the data and computer used; however, for this study, we set this value to 10, which is the same as the number of CPU cores.

5.1 Synthetic datasets

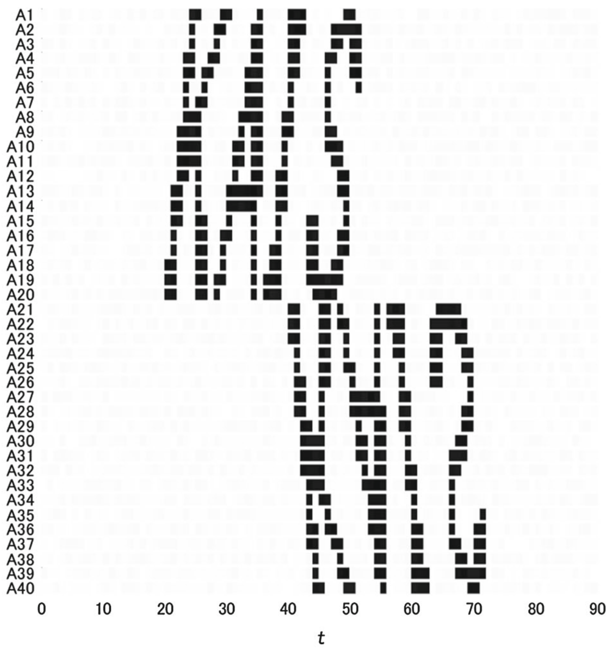
We conducted the first experiment using two synthetic datasets. We made the two datasets where a 20-dimensional EP obviously exists. Figure 7 shows the two datasets, which we call synthetic datasets A and B.

Synthetic dataset A has 40 sequences and the IDs are A1,..., A40. Synthetic dataset B has 20 sequences and the IDs are B1,..., B20. The length of a sequence in synthetic dataset A is 90, and that in B is 80. A pattern appears from $t = 20$ in sequences whose IDs are A1,..., A20; the same pattern appears from $t = 40$ in sequences whose IDs are A40,..., A21. In addition, the same pattern appears from $t = 30$ in synthetic dataset B; however, the order of the sequences is out of order.

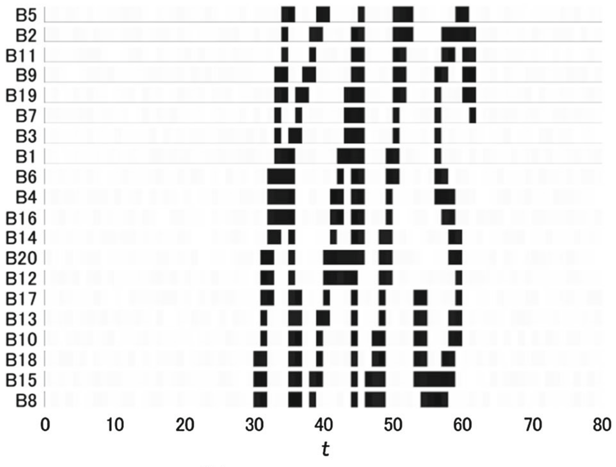
We normalized all sequences such that the means and standard deviations would be 0 and 1, respectively. We used the similarity function s described in Sect. 2.1 to determine whether the two multidimensional sequences were equivalent. We set the length τ of the subsequences and the threshold θ_{MSV} of MSV to 20 and 0.01, respectively, and two multidimensional sequences were determined equivalent if the value of s was greater than 0.99.

Table 3 shows the processing times for PIS and PDFS. Processing times for PIS and PDFS were 58 min 18 s and 1 min 29 s, respectively, which means that PDFS was 39 times faster than PIS. The processing times required to obtain two-dimensional EPs were 1 min 24 s for both methods; the processing times to obtain EPs with three or more dimensions were 56 min 53 s for PIS and 5 s for PDFS, thereby indicating PDFS was 648 times faster when the number K of dimensions was greater than two. Since EPs with different numbers of dimensions may be searched in parallel in PDFS, only the total processing time to obtain EPs with three or more dimensions was measured. Although it is possible to measure the processing time to obtain EPs in each dimension in PDFS, adding the processing times to obtain EPs in each dimension does not yield the total processing time in general. In PIS, adding the processing times to obtain EPs in each dimension yields the total processing time.

Figure 8 shows the numbers of comparisons of sequences in each dimension K of EPs and the numbers of EPs obtained in PIS and PDFS. When $K = 2$, the number of comparisons for PIS and that for PDFS were the same; however, when $K = 3$ to 19, the number of



(a) Synthetic dataset 1



(b) Synthetic dataset 2

Fig. 7 Synthetic datasets. Black points denote values of 1; white points denote values of 0

comparisons for PIS was significantly higher, and a combinatorial explosion occurred in PIS; however, it did not occur in PDFS. When $K = 20$, the number of comparisons was two for both methods. The number of two-dimensional EPs obtained by PDFS was the same as that by PIS; however, the number of 3- to 19-dimensional EPs obtained by PDFS was significantly smaller than that by PIS, and the number of 20-dimensional EPs was two for both methods. Of the EPs obtained, there were only two NDEPs in 20 dimensions, and the

Table 3 Processing times for synthetic datasets (min:sec). K is the number of dimensions

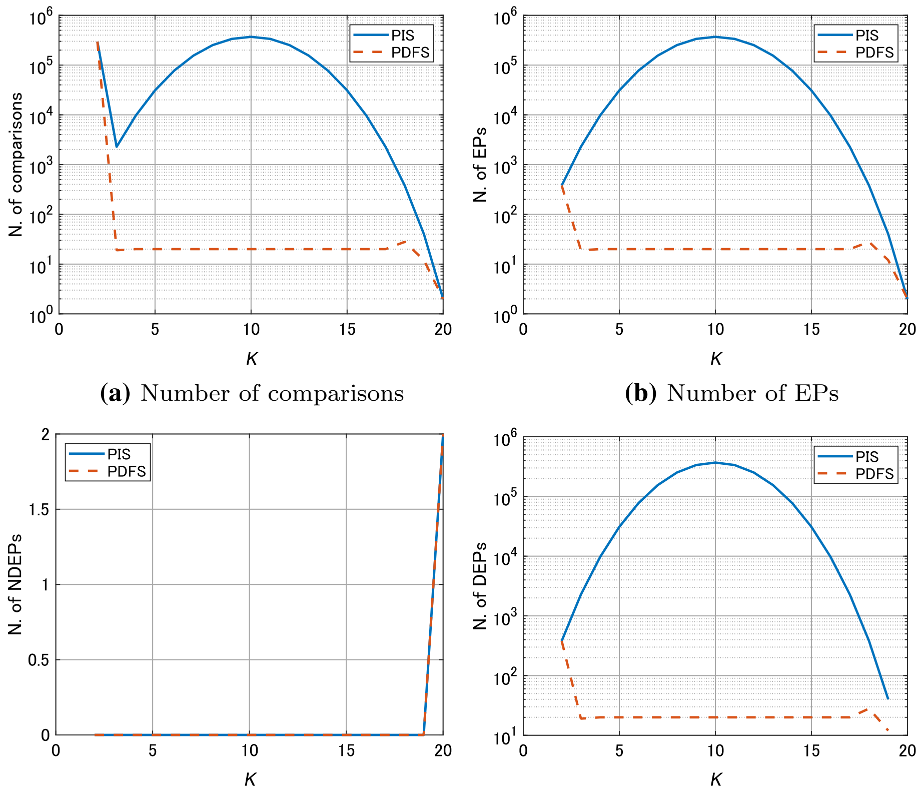
K	PIS	PDFS
2	01:24	01:24
3	00:01	N/A
4	00:06	N/A
5	00:25	N/A
6	01:15	N/A
7	02:54	N/A
8	05:21	N/A
9	08:08	N/A
10	09:55	N/A
11	09:56	N/A
12	08:17	N/A
13	05:34	N/A
14	03:01	N/A
15	01:19	N/A
16	00:27	N/A
17	00:07	N/A
18	00:01	N/A
19	00:00	N/A
20	00:00	N/A
3–20	56:53	00:05
Total	58:18	01:29

number of NDEPs obtained by PIS and that by PDFS were the same. One of the reasons why the number of comparisons for PIS and that for PDFS were the same when $K = 20$ is that comparisons to obtain NDEPs in PDFS cannot be skipped, unlike comparisons to obtain DEPs. The number of DEPs obtained by PIS was the same as that by PDFS when $K = 2$; however, the number of DEPs obtained by PIS was significantly higher than that by PDFS when $K \geq 3$. When $K = 10$, the number of DEPs obtained by PIS was 369500, which was the highest; however, the number of DEPs in PDFS was 20. The number of DEPs obtained by PDFS was always dozens when $K = 3$ or more. The number of DEPs obtained by PDFS will change if the parameter n_{para} is changed.

The maximum number of dimensions of the obtained EPs was 20, and 20-dimensional EPs obtained by PDFS were exactly like those obtained by PIS. Table 4 lists the 20-dimensional EPs. Based on the arrangements of the sequences in Fig. 7, $E_1^{(20)}$ and $E_2^{(20)}$ in Table 4, it seems that the correspondence relation was successfully identified; however, ES extraction does not enable us to find similar patterns themselves. If such patterns are necessary, we can use motif discovery methods after finding dimensions corresponding between two datasets.

5.2 Motion capture datasets

Using two motion capture datasets, we conducted the same experiment as reported in a previous study [9] (data, preprocessing, and parameter values were the same). The only difference was the computer used and the version of MATLAB. We obtained the datasets



(a) Number of comparisons **(b)** Number of EPs **(c)** Number of NDEPs **(d)** Number of DEPs

Fig. 8 Numbers of comparisons and EPs for synthetic datasets

Table 4 Highest-dimensional EPs obtained by PIS for synthetic datasets

$E*_1^{(20)}$	{⟨	A1,	A2,	A3,	A4,	A5,	A6,	A7,	A8,	A9,	A10,
		A11,	A12,	A13,	A14,	A15,	A16,	A17,	A18,	A19,	A20⟩,
	⟨	B5,	B2,	B11,	B9,	B19,	B7,	B3,	B1,	B6,	B4,
		B16,	B14,	B20,	B12,	B17,	B13,	B10,	B18,	B15,	B8⟩}
$E*_2^{(20)}$	{⟨	A21,	A22,	A23,	A24,	A25,	A26,	A27,	A28,	A29,	A30,
		A31,	A32,	A33,	A34,	A35,	A36,	A37,	A38,	A39,	A40⟩,
	⟨	B8,	B15,	B18,	B10,	B13,	B17,	B12,	B20,	B14,	B16,
		B4,	B6,	B1,	B3,	B7,	B19,	B9,	B11,	B2,	B5⟩}

(07_01.c3d, 07_02.c3d) from CMU Graphics Lab Motion Capture Database. Here, we call the two datasets mocap dataset A and B. Each dataset has 41 sequences; however, because there are sequences similar to other sequences, these sequences were merged before ES extraction. After merging sequences, the number of sequences in mocap dataset A was 22, and that in mocap dataset B was 18. Next, the original length of the sequences in mocap dataset A is 316 and that in mocap dataset B is 329; however, using simple moving average

Table 5 Processing times for mocap datasets (min:sec)

K	PIS	PDFS
2	00:02	00:03
3	00:00	N/A
4	00:01	N/A
5	00:04	N/A
6	00:07	N/A
7	00:10	N/A
8	00:11	N/A
9	00:11	N/A
10	00:07	N/A
11	00:03	N/A
12	00:01	N/A
13	00:00	N/A
14	00:00	N/A
15	00:00	N/A
3–15	01:00	00:36
Total	01:03	00:39

and downsampling as preprocessing, the length for mocap dataset A was 52, and that for mocap dataset B was 54. As in the previous study [9], we set the length τ of subsequences and the threshold θ_{MSV} of MSV to 20 and 0.06, respectively, as used in the similarity function s described in Sect. 2.1; the sequences were equivalent if the value of s was less than 0.05.

Table 5 shows the processing times for PIS and PDFS. They were 1 min 3 s and 39 s, respectively, which means that PDFS was 1.6 times faster. The processing times to obtain two-dimensional EPs were almost the same for both methods; however, the processing times to obtain EPs with three or more dimensions were 1 min 0 s for PIS and 36 s for PDFS, which means that PDFS was 1.7 times faster. Since EPs with different numbers of dimensions may be searched in parallel, we measured only the processing time to obtain EPs with three or more dimensions in the procedure of PDFS. Figure 9 shows the numbers of comparisons of sequences in each dimension K of EPs and the numbers of EPs obtained in PIS and PDFS.

Figure 9 shows the numbers of comparisons of sequences in each dimension K of EPs and the numbers of EPs obtained in PIS and PDFS. When $K = 2$, the number of comparisons for PIS and that for PDFS were the same; however, when $K = 3$–14, the number of comparisons for PIS was higher. When $K = 15$, the number of comparisons for PIS and that for PIS were the same.

As shown in Fig. 9b, the number of two-dimensional EPs obtained by PDFS was the same as that by PIS; however, the number of 3- to 14-dimensional EPs obtained by PDFS was significantly smaller than that by PIS. The number of 15-dimensional EPs was 6 for both PDFS and PIS.

In the experiment using synthetic datasets, the number of NDEPs obtained by PIS was the same as that by PDFS; however, in this experiment, more NDEPs were obtained by PIS as shown in 9c. For example, when $K = 14$, the number of NDEPs obtained by PIS was 31, and that obtained by PDFS was 29. Thus, for examining NDEPs obtained only by PIS, it was found that not all DEPs were obtained. For example, EP $\{\langle$A6, A7, A8, A9, A10, A12, A14, A15, A16, A18, A19, A20, A21, A22$\rangle$, $\langle$B6, B9, B3, B11, B5, B16, B12, B15, B10,

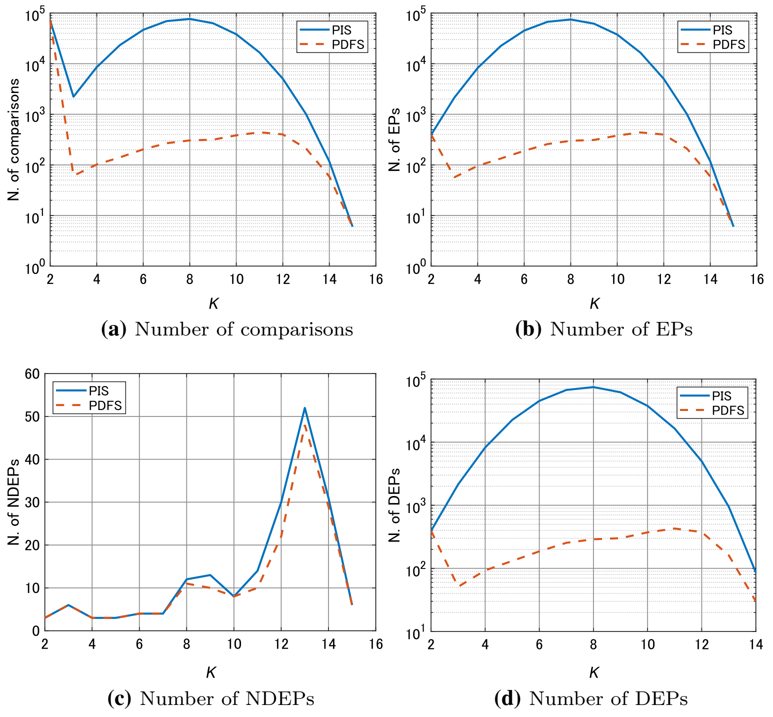


Fig. 9 Numbers of comparisons and EPs for mocap datasets

B1, B17, B2, B18, B8⟩ } were obtained only by PIS, and some of the DEPs such as { ⟨A7, A9, A10⟩, ⟨B9, B11, B5⟩ }, { ⟨A7, A14, A16⟩, ⟨B9, B12, B10⟩ }, and { ⟨A9, A10, A14⟩, ⟨B11, B5, B12⟩ } were not obtained even when using PIS. If Assumption 1 always holds, then all DEPs should be obtained by PIS, but this is not the case. We believe that this is more likely to occur in noisy datasets. Further, we consider that it is not necessary to obtain all NDEPs, and it is sufficient to obtain *useful* NDEPs. Currently, there is no standard to evaluate the usefulness of an NDEP, and here, we consider that an EP with more dimensions is more useful.

The maximum dimension of EPs obtained was 15, and 15-dimensional EPs obtained by PDFS were the same as those obtained by PIS. Table 6 shows the 15-dimensional EPs. Figure 10 shows 15-dimensional EP $E_4^{(15)}$ drawn in the original three-dimensional space. From the results shown in Fig. 10, we consider that quality correspondence relations were successfully determined.

5.3 Video datasets

Using two movie datasets, we conducted an experiment to determine the correspondence relations between the two movies. In this experiment, we used two movies (one was embedded in the other video), and we verified whether the embedded video could be found by ES extraction. One application of this experiment is to search for illegally uploaded videos. Embedding an illegally uploaded video to another video makes finding the illegally uploaded video very difficult. However, even if such embedding is performed, we consider that, by using ES extraction, it is possible to discover whether the target movie is embedded. Because

Table 6 Highest-dimensional EPs obtained by PIS or PDFS for mocap datasets

$E{*}_1^{(15)}$	{ ⟨	A1,	A6,	A7,	A9,	A10,	A12,	A14,	A15,
		A16,	A17,	A18,	A19,	A20,	A21,	A22⟩,	
	⟨	B4,	B6,	B9,	B11,	B5,	B16,	B12,	B14,
		B10,	B15,	B1,	B17,	B2,	B18,	B8⟩ }	
$E{*}_2^{(15)}$	{ ⟨	A4,	A6,	A7,	A8,	A9,	A10,	A12,	A15,
		A16,	A17,	A18,	A19,	A20,	A21,	A22⟩,	
	⟨	B4,	B6,	B9,	B3,	B11,	B5,	B16,	B14,
		B10,	B15,	B1,	B17,	B2,	B18,	B8⟩ }	
$E{*}_3^{(15)}$	{ ⟨	A4,	A6,	A7,	A8,	A9,	A10,	A12,	A15,
		A16,	A17,	A18,	A19,	A20,	A21,	A22⟩,	
	⟨	B4,	B6,	B9,	B3,	B11,	B5,	B16,	B15,
		B10,	B14,	B1,	B17,	B2,	B18,	B8⟩ }	
$E{*}_4^{(15)}$	{ ⟨	A4,	A6,	A8,	A9,	A10,	A11,	A12,	A15,
		A16,	A17,	A18,	A19,	A20,	A21,	A22⟩,	
	⟨	B4,	B6,	B3,	B11,	B5,	B9,	B16,	B14,
		B10,	B15,	B1,	B17,	B2,	B18,	B8⟩ }	
$E{*}_5^{(15)}$	{ ⟨	A4,	A6,	A8,	A9,	A10,	A11,	A12,	A15,
		A16,	A17,	A18,	A19,	A20,	A21,	A22⟩,	
	⟨	B4,	B6,	B3,	B11,	B5,	B9,	B16,	B15,
		B10,	B14,	B1,	B17,	B2,	B18,	B8⟩ }	
$E{*}_6^{(15)}$	{ ⟨	A6,	A7,	A8,	A9,	A10,	A12,	A14,	A15,
		A16,	A17,	A18,	A19,	A20,	A21,	A22⟩,	
	⟨	B6,	B9,	B3,	B11,	B5,	B16,	B12,	B14,
		B10,	B15,	B1,	B17,	B2,	B18,	B8⟩ }	

we attempted to compare PIS and PDFS, we did not use a method specific to this problem. There are probably many ways to specialize it to this problem. For example, using information on which sequences are adjacent can help improve speed; however, this information was not used in this experiment.

We used two videos from Videvo. The titles of the two videos are *Beetle in Slow Motion CC-BY NatureClip*, and *Car Journey Time-Lapse at Night CC-BY NatureClip*. Here, we refer to them as videos 1 and 2, respectively. Figure 11 shows one frame for each video.

Here, we embedded video 1 in video 2 and verified whether the embedded video could be identified. The process for embedding video 1 in video 2 and creating two datasets is shown below.

1. The frame size (width $\times$ height) of video 1 is 1280×720; we resized it to 6×5 (Fig. 12a).
2. Let video 1 be a dataset of 30 sequences. Here, we call this dataset video dataset A.
3. The frame size (width $\times$ height) of video 2 is 1280×720; we resized it to 10×10 (Fig. 12b).
4. We rotated video 1 by 90 degree and embedded it in video 2 as shown in Fig. 12c; we played video 1 after 5 s as shown in Fig. 12d.
5. Let video 2 be a dataset of 100 sequences. Here, we call this dataset video dataset B.

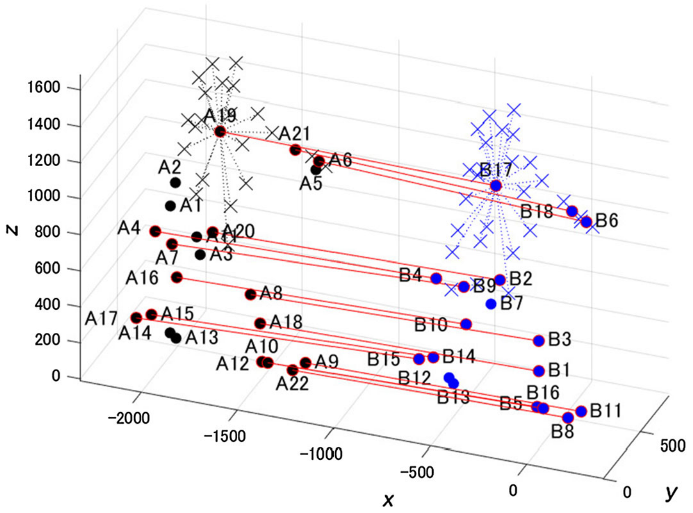
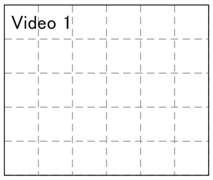


Fig. 10 15-dimensional EP $E_4^{(15)}$ drawn in the original three-dimensional space. Circle markers represent sequences for IDs A1,..., A22, B1,..., B18. Cross markers represent sequences that were merged in the preprocessing. Solid lines represent the correspondence relation

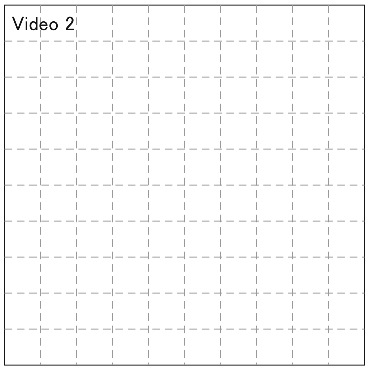
Fig. 11 Videos

As preprocessing for ES extraction, we normalized each sequence such that the mean and standard deviation would be 0 and 1, respectively. Next, we set n to six and used the n-point simple moving average for each sequence. We selected the value of n such that the mean squared error between sequences before and after the moving average was 0.01 or less in all sequences in video dataset A. Then, we down-sampled each sequence by n. Because the original frame per sec (FPS) was 24, the FPS after this processing was 4.

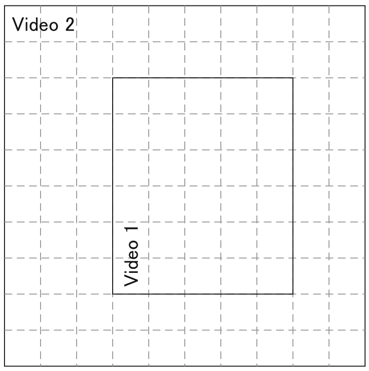
As in the other experiments, we used the similarity function s described in Sect. 2.1 to determine whether the two multidimensional sequences were equivalent. We set the length τ of the subsequences and the threshold θ_{MSV} of MSV to 20 and 0.1, respectively, and two



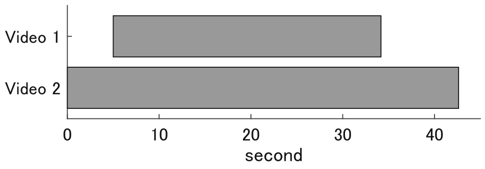
(a) Video 1 (5 × 6). We call this dataset video dataset A.



(b) Video 2 (10 × 10)



(c) Embedment of video 1 in video 2. We call this dataset video dataset B.



(d) Time axis of video dataset B. We embedded video 1 in video 2 and played it after 5 sec.

Fig. 12 Video datasets

multidimensional sequences were determined equivalent if the value of s was greater than 0.5.

Table 7 summarizes the processing times for PIS and PDFS. When searching for nine-dimensional EPs in PIS, memory usage reached its limit, and therefore, the process ended when $K = 9$. The processing time up to $K = 8$ was 448 min 46 s. The memory was not out of memory in the procedure of PDFS, and the processing time up to $K = 29$ was 84 min 21 s, which means that the total processing time for PDFS was 6.3 times faster than the processing time for PIS up to $K = 8$. The processing times to obtain two-dimensional EPs were about 84 min for both methods. In PDFS, the processing time from $K =$ 3–29 was 22 s, and the processing time from $K =$ 3–8 in PIS was 532 min 43 s, which means that the processing time from $K =$ 3–29 in PDFS was 1203 times faster than that from $K =$ 3–8 in PIS.

Table 7 Processing times for video datasets (min:sec)

K	PIS	PDFS
2	83:57	83:59
3	00:07	N/A
4	00:54	N/A
5	05:27	N/A
6	25:24	N/A
7	94:55	N/A
8	321:57	N/A
9	out of memory	N/A
3–8	448:46	N/A
3–29	N/A	00:22
Total	532:43	84:21

Figure 13 shows the numbers of comparisons of sequences in each dimension K of EPs and the numbers of EPs obtained in PIS and PDFS. Since PIS ends processing in the middle, the figures show the numbers of NDEPs and DEPs obtained only by PDFS. When $K = 2$, the number of comparisons for PIS and that for PDFS were the same; however, a combinatorial explosion occurred in PIS when $K > 2$. The number of comparisons for PDFS did not exceed 100 when K was greater than three. In addition, the number of EPs obtained by PIS increased explosively, and that by PDFS did not exceed 100. The number of NDEPs obtained by PDFS was 0 to 3 when $K = 2, ..., 29$. The total number of the NDEPs was 11, and the maximum number of dimensions of NDEPs was 29. Figure 14 shows the 29-dimensional EP obtained by PDFS. Video dataset A has 30 sequences of video 1, and the 30 sequences were embedded in video dataset B. As shown in Fig. 14, 29 out of 30 sequences of video 1 were successfully found in video dataset B. We determined that sequences were equivalent if the value of s was less than 0.5 in this experiment. The 30-dimensional EP might be obtained by increasing the value to more than 0.5; however, in that case, extra EPs may be obtained, which can lead to a slow process. In addition, the larger the size of the images, the more accurate the distinctiveness may be. In order to handle a larger size of images, it will be necessary to use more powerful computers or a faster method to obtain two-dimensional EPs.

6 Conclusion

In this paper, we showed that PIS—the fastest of the existing methods—does indeed have a problem in that the number of unnecessary EPs causes a combinatorial explosion. Further, we proposed a new method, PDFS, which can avoid any combinatorial explosion. In the experiment using synthetic datasets, the processing times for PIS and PDFS were 58 min 18 s and 1 min 29 s, respectively, and thus, PDFS was 39 times faster than PIS. The maximum number of dimensions of the obtained EPs was 20, and two 20-dimensional EPs obtained by PDFS were exactly the same as those obtained by PIS. In the experiment using motion capture datasets, the processing times for PIS and PDFS were 1 min 3 s and 39 s, respectively; thus, PDFS was 1.6 times faster than PIS. The maximum number of dimensions of the obtained EPs was 15, and six 15-dimensional EPs obtained by PDFS were exactly the same as those obtained by PIS. In the experiment using video datasets, the process of PIS was terminated

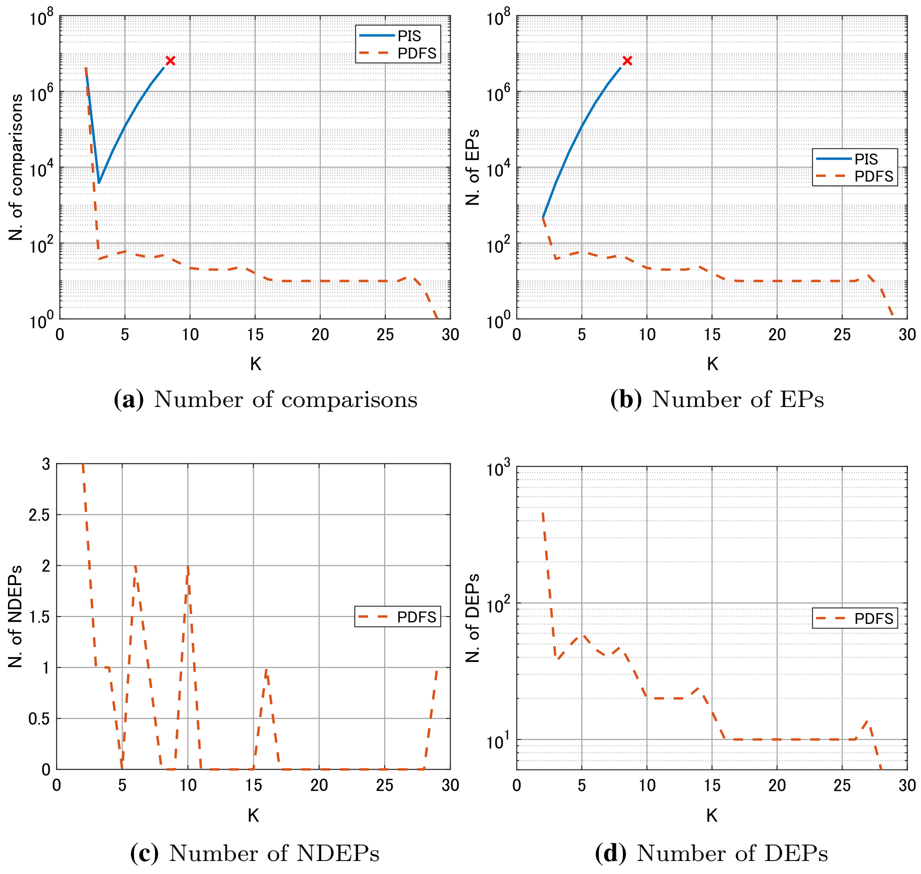


Fig. 13 Numbers of comparisons and EPs for video datasets

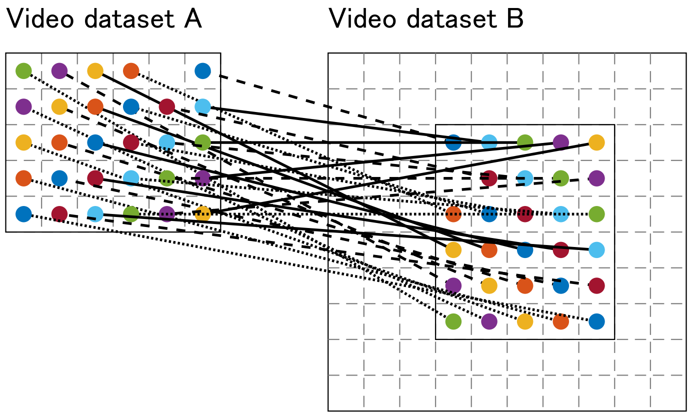


Fig. 14 29-dimensional EP obtained by PDFS. Lines represent the correspondence relation

because memory usage reached its limit while searching for nine-dimensional EPs. The memory was not insufficient for PDFS; further, PDFS enabled us to obtain a 29-dimensional EP that implied a quality correspondence relation between two video datasets. The processing time for PDFS up to 29 dimensions was 84 min 21 s, and that for PIS up to 8 dimensions was 532 min 43. This means the total processing time for PDFS was 6.3 times shorter than that for PIS even up to 8 dimensions. In addition, the processing time for PDFS from 3–29 dimensions was 22 s, and the processing time for PIS from 3–8 dimensions was 448 min 46 s. This means the processing time for PDFS from 3–29 dimensions was 1203 times faster than that for PIS from 3–8 dimensions.

In the future, we will propose a faster method to obtain all two-dimensional EPs because it usually takes a considerable amount of time to obtain all two-dimensional EPs even when using PDFS. In addition, we plan to conduct more practical application experiments such as the analysis of time series data, preprocessing of imitation learning and preprocessing of transfer learning as described in Sect. 1. Moreover, because there has been no standard to evaluate the usefulness of an EP, such a standard will be useful. If we have such standard to evaluate the usefulness of an EP, then a similarity function can be improved.

References

1. Alaee S, Kamgar K, Keogh E (2020) Matrix profile XXII: exact discovery of time series motifs under DTW. In: Int Conf on Data Mining (ICDM)
2. Caselles-Dupré H, Ortiz MG, Filliat D (2019) Symmetry-based disentangled representation learning requires interaction with environments. Adv Neural Inf Process Syst 32:4606
3. Delhaisse B, Esteban D, Rozo L, Caldwell D (2017) Transfer learning of shared latent spaces between robots with similar kinematic structure. In: Int Joint Conf on Neural Networks (IJCNN)
4. Gao Y, Lin J (2019) Discovering subdimensional motifs of different lengths in large-scale multivariate time series. In: Int Conf on Data Mining (ICDM)
5. Higgins I, Amos D, Pfau D, Racaniere S, Matthey L, Rezende D, Lerchner A (2018) Towards a definition of disentangled representations. arXiv:1812.02230
6. Locatello F, Poole B, Rätsch G, Schölkopf B, Bachem O, Tschannen M (2020) Weakly-supervised disentanglement without compromises. In: Int Conf on Machine Learning (ICML)
7. Long M, Zhu H, Wang J, Jordan MI (2017) Deep transfer learning with joint adaptation networks. In: Int Conf on Machine Learning (ICML)
8. Satoh S, Takahashi Y, Yamakawa H (2017) Validation of equivalence structure incremental search. Front Robot AI. https://doi.org/10.3389/frobt.2017.00063
9. Satoh S, Takahashi Y, Yamakawa H (2018) Accelerated equivalence structure extraction via pairwise incremental search. In: ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining
10. Satoh S, Yamakawa H (2017) Incremental extraction of high-dimensional equivalence structures. In: Int Joint Conf on Neural Networks (IJCNN)
11. Senin P (2008) Dynamic time warping algorithm review. Tech rep, Information and Computer Science Department University of Hawaii at Manoa Honolulu USA
12. Sermanet P, Lynch C, Chebotar Y, Hsu J, Jang E, Schaal S, Levine S, Brain G (2018) Time-contrastive networks: self-supervised learning from video. In: Int Conf on Robotics and Automation (ICRA)
13. Sun Q, Liu Y, Chua TS, Schiele B (2019) Meta-transfer learning for few-shot learning. In: Conf on Computer Vision and Pattern Recognition (CVPR)
14. Torabi F, Warnell G, Stone P (2019) Recent advances in imitation learning from observation. In: Int Joint Conf on Artificial Intelligence (IJCAI)
15. Wang J, Chen Y, Feng W, Yu H, Huang M, Yang Q (2020) Transfer learning with dynamic distribution adaptation. ACM Trans Intell Syst Technol 11(1):1–25
16. Yamada M, Kim H, Miyoshi K, Iwata T, Yamakawa H (2020) Disentangled representations for sequence data using information bottleneck principle. In: Asian Conf on Machine Learning (ACML)
17. Yang C, Yuan K, Heng S, Komura T, Li Z (2020) Learning natural locomotion behaviors for humanoid robots using human bias. IEEE Robot Autom Lett 5(2):2610–2617

18. Yeh CCM, Kavantzas N, Keogh E (2017) Matrix profile VI: meaningful multidimensional motif discovery. In: Int Conf on Data Mining (ICDM)
19. Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, Xiong H, He Q (2019) A comprehensive survey on transfer learning. arXiv:1911.02685

Seiya Satoh is currently an Assistant Professor in the School of Science and Engineering at Tokyo Denki University in Japan. He received the Ph.D. degree in computer science from Chubu University in Japan. His research interests include machine learning and data mining.

Hiroshi Yamakawa is currently a Chairperson of The Whole Brain Architecture Initiative (WBAI), non-profit organization, a Visiting Professor of the University of Electro-Communications and a Visiting Professor of the Kindai University. He received an M.S. in physics and Ph.D. in engineering from the University of Tokyo in 1989 and 1992 respectively. His research interest is brain-inspired artificial general intelligence and concept formation.