PAPER

Measuring the Similarity of Protein Structures Using Image Compression Algorithms

Morihiro HAYASHIDA[†a)] ***and*** **Tatsuya AKUTSU**[†], ***Members***

SUMMARY For measuring the similarity of biological sequences and structures such as DNA sequences, protein sequences, and tertiary structures, several compression-based methods have been developed. However, they are based on compression algorithms only for sequential data. For instance, protein structures can be represented by two-dimensional distance matrices. Therefore, it is expected that image compression is useful for measuring the similarity of protein structures because image compression algorithms compress data horizontally and vertically. This paper proposes series of methods for measuring the similarity of protein structures. In the methods, an original protein structure is transformed into a distance matrix, which is regarded as a two-dimensional image. Then, the similarity of two protein structures is measured by a kind of compression ratio of the concatenated image. We employed several image compression algorithms, JPEG, GIF, PNG, IFS, and SPC. Since SPC often gave better results among the other image compression methods, and it is simple and easy to be modified, we modified SPC and obtained MSPC. We applied the proposed methods to clustering of protein structures, and performed Receiver Operating Characteristic (ROC) analysis. The results of computational experiments suggest that MSPC has the best performance among existing compression-based methods. We also present some theoretical results on the time complexity and Kolmogorov complexity of image compression-based protein structure comparison.

key words: *image compression, universal similarity metric, protein structure comparison*

1. Introduction

Understanding of protein structures is one of the important topics in bioinformatics and computational biology. In particular, classification of protein structures is important and thus many studies have been done and several databases have been developed such as SCOP [1] and CATH [2]. Classification of protein structures is usually done based on some measure of the similarity between protein structures.

However, an agreement on which is the best similarity measure is not yet obtained and a variety of structure comparison methods have been proposed. Most existing methods are based on *protein structure alignment*. Various methodologies have been employed for protein structure alignment, which include double dynamic programming [3], iterative improvement [4], combinatorial extension [5], comparisons of distance matrices [6], use of partial order graphs [7], contact map overlap [8], artificial neural networks using the convex hull representation of a protein structure [9], grouping atoms for interaction sites [10], and partial geometric hashing [11]. In most of structure alignment methods, some scoring function is defined for measuring the quality of the obtained alignment. Then, the structure alignment problem is defined as finding a structure alignment with the optimal or near optimal score. However, score functions are defined in more or less ad-hoc manners and there is no consensus or theoretical justification. Furthermore, many of existing structure alignment methods are not very efficient.

Shibuya et al. developed a linear-time protein structure searching algorithm [12]. However, their problem of reporting all substructures similar to a query structure from a database has been proved to be NP-hard, and it is difficult to solve the problem if there are many insertions and deletions.

Krasnogor and Pelta [13], Barthel et al. [14], and Shah et al. [15] proposed a novel approach to measuring the similarity of protein structures, and developed the decision support system ProCKSI for protein structure comparison and other tasks. Their method is similar to the contact map overlap (CMO) approach [8]. In their method, each protein structure is transformed into a 0-1 matrix, which is further regarded as a 0-1 sequence. Then, two protein structures are compared based on the compression ratio of the sequence obtained by concatenating two 0-1 sequences. They approximated Kolmogorov complexities by the compression ratios, and calculated the universal similarity metric (USM) proposed by Li et al. [16]. Their method is quite simple to implement and very fast. They demonstrated the usefulness of the method by means of application to clustering of protein structures. It is worthy to mention that several works have been done on measuring the similarity of biological sequences based on data compression approach [16], [17].

Though the approach by Krasnogor and Pelta is novel and useful, the distances between residues are truncated into 0 or 1. As a result, the similarity measure depends on the threshold, which should be determined by trial and error. Pelta et al. and Terrazas et al. also addressed this problem [18], [19]. The same drawback applies to CMO [8].

Ferragina et al. proposed three similarity measures by approximating the USM, that is, UCD (Universal Compression Dissimilarity), NCD (Normalized Compression Dissimilarity), and CD (Compression Dissimilarity) [20]. They experimentally tested these measures by using 25 existing compressors for some kinds of biological data such as DNA sequences, protein sequences, protein secondary and tertiary structures. They concluded that the combination of UCD or

Manuscript received March 14, 2011.
Manuscript revised July 13, 2011.

[†]The authors are with Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji-shi, 611–0011 Japan.
a) E-mail: morihiro@kuicr.kyoto-u.ac.jp

DOI: 10.1587/transinf.E94.D.2468

NCD and UPGMA (Unweighted Pair Group Method with Arithmetic Mean) should be used, and as for compressors, PPMd [21], [22] and Gencompress [23] are able to give the best results in most cases.

On the other hand, similarity measures based on protein sequence comparison have been studied. Dai and Wang proposed similarity measures using protein *sequence space* [24]. They obtained 20 sets of similar amino acids, called star sets, according to the substitution matrix such as BLOSUM62 [25] and PAM40 [26], and defined the sequence space for a protein sequence to be the set of all the sequences obtained by replacing each amino acid of the protein sequence with an element of the star set. Then, dissimilarity measures are calculated as distances using frequencies of k consecutive amino acids. They reported that the sequence space is able to provide more information than the protein sequence only and contributes to the classification accuracy to classes of CATH, especially for less redundant datasets. However, the accuracy is not considered to be sufficient for practical use, and it is considered that there is a limit on the classification using only protein sequences.

In this paper, we try to overcome such drawbacks using the compression based approach and a very simple idea. We employ *image compression* in place of sequence compression. It is expected that two-dimensional relationships that are not obtained by only sequence compression are obtained by image compression. Each distance matrix (not 0-1 matrix) is directly compressed by using an image compression algorithm. We examine the following image compression algorithms: JPEG, GIF, PNG, IFS, and SPC. We find that the classification results by SPC is the best among these compression algorithms. However, in a simple application of image compression algorithms, the concatenated image may include an extra region because images are usually represented as rectangular shapes. Therefore, we propose a methodology to avoid use of the extra region by modifying SPC. We apply the proposed methods to clustering and classification of protein structures as in [13], [20], [24].

The organization of the paper is as follows. We begin with a brief review the methods by Krasnogor and Pelta [13] and Ferragina et al. [20]. Next, we present our proposed methods. Then, we describe details and results of computational experiments, and provide several theoretical results. Finally, we conclude with future work.

2. Methods

2.1 Structure Comparison Using Sequence Compression

Krasnogor and Pelta [13] and Ferragina et al. [20] employed sequence compression to measure the similarity of two proteins. Their methods are based on the universal similarity metric (USM), which was originally proposed by Li et al. [16]. USM is based on Kolmogorov complexity. The Kolmogorov complexity $K(o)$ of an object o is defined to be the length of the shortest program P for a Universal Turing Machine U that is required to output o [27]. That is, $K(o)$ is defined by $K(o) = \min\{|P| \mid P \text{ is a program such that } U(P) = o\}$, and is considered to be a measure of the amount of information contained in o. Besides, the conditional Kolmogorov complexity of o_1 given o_2 is defined by $K(o_1|o_2) = \min\{|P| \mid P \text{ is a program such that } U(P, o_2) = o_1\}$, where $U(P, o_2) = o_1$ means that program P outputs o_1 when o_2 is given. Based on these, information distance between two objects o_1 and o_2 can be defined as $\max\{K(o_1|o_2), K(o_2|o_1)\}$. Since this distance is not normalized, USM was proposed as a normalized measure [16]:

$$USM(o_1, o_2) = \frac{\max\{K(o_1|o_2), K(o_2|o_1)\}}{\max\{K(o_1), K(o_2)\}}.$$

It is well-known that Kolmogorov complexity of a given object is not computable [27]. Thus, Krasnogor and Pelta, and Ferragina et al. employed sequence compression algorithms such as ‘compress’, ‘gzip’, and ‘bzip2’ commands in UNIX. Let $C(s)$ be the size of the compressed sequence of s. They used $C(o_1)$ and $C(o_1 \cdot o_2) - C(o_2)$ in place of $K(o_1)$ and $K(o_1|o_2)$ respectively, where $o_1 \cdot o_2$ denotes the concatenation of two sequences o_1 and o_2. Ferragina et al. used protein amino acid sequences, TOPS string of secondary structure elements with and without the contact information [28], and ATOM lines from the PDB entry [29] as the sequences o_k for protein P_k. They proposed three kinds of similarity measures, UCD (Universal Compression Dissimilarity), NCD (Normalized Compression Dissimilarity), and CD (Compression Dissimilarity) by approximating USM with the size of compressed sequence $C(s)$ as follows:

$$\begin{aligned}
&UCD(o_1, o_2)\\
&\quad= \frac{\max\{C(o_1 \cdot o_2) - C(o_1), C(o_2 \cdot o_1) - C(o_2)\}}{\max\{C(o_1), C(o_2)\}},\\
&NCD(o_1, o_2)\\
&\quad= \frac{\min\{C(o_1 \cdot o_2), C(o_2 \cdot o_1)\} - \min\{C(o_1), C(o_2)\}}{\max\{C(o_1), C(o_2)\}},\\
&CD(o_1, o_2)\\
&\quad= \frac{\min\{C(o_1 \cdot o_2), C(o_2 \cdot o_1), C(o_1) + C(o_2)\}}{C(o_1) + C(o_2)}.
\end{aligned}$$

On the other hand, Krasnogor and Pelta used a 0-1 sequence obtained from a contact map M_k of protein structure P_k as the sequence o_k, where $M_k[i, j] = 1$ if the distance between the C_α atom of ith residue and that of jth residue is less than threshold θ, otherwise $M_k[i, j] = 0$. o_k is obtained by simple raster scanning of matrix M_k.

2.2 Similarity Metric Based on Image Compression

We define a contact map M_k of protein P_k as the distance matrix between residues as $M_k[i, j] = \sqrt{(r_k[i] - r_k[j])^2}$, where $r_k[i]$ denotes the three-dimensional coordinate of ith C_α atom of P_k.

We transform the contact map M_k to a raw image format, PPM (Portable Pixel Map). PPM can represent $(2^8)^3 =$ 16777216 colors using 3 bytes memory for a pixel, where each byte is used for red, green, and blue, respectively, zero

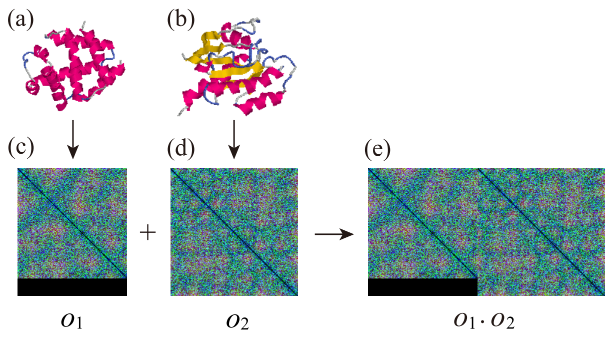


Fig. 1 Examples of images and concatenated images for proteins (a) 1ash and (b) 1aa9. (c) The image o_1 of 1ash, which is filled with black color to the height of 1aa9. (d) The image o_2 of 1aa9. (e) The concatenated image $o_1 \cdot o_2$ of 1ash and 1aa9.

means black color, and $16777215 (= (2^8)^3 - 1)$ means white color. We transform $M_k[i, j]$ to the corresponding pixel with the color of the integer part of $cM_k[i, j]$, where c is a constant, and we set $c = 4 \cdot (2^8)^2 = 262144$ in the experiment section. The upper, middle, and lower bytes of $cM_k[i, j]$ are set as red, green, and blue, respectively. If $cM_k[i, j]$ is greater than or equal to $(2^8)^3 = 16777216$, we set the color white. Almost all values of $cM_k[i, j]$ for the evaluated proteins are less than the value $(2^8)^3$. Figures 1 (c) and (d) show examples of such images for proteins 1ash and 1aa9 (Figs. 1 (a) and (b)), respectively. In order to concatenate two images horizontally, the two images must have the same height. Therefore, we fill the smaller image with black color to the height of the other (See Figs. 1 (c) and (e)).

Krasnogor and Pelta approximated $K(o_1|o_2)$ of USM by $C(o_1 \cdot o_2) - C(o_2)$ [13]. However, $C(o_1 \cdot o_2)$ is not always equal to $C(o_2 \cdot o_1)$. Therefore, we approximate $K(o_1|o_2)$ by $\max\{C(o_1 \cdot o_2) - C(o_2), C(o_2 \cdot o_1) - C(o_2)\} = \max\{C(o_1 \cdot o_2), C(o_2 \cdot o_1)\} - C(o_2)$. Then, the approximated USM for image compression, ACD (Anticommutative Compression Dissimilarity), is given as follows:

$$ACD(o_1, o_2) = \frac{\max\{C(o_1 \cdot o_2), C(o_2 \cdot o_1)\} - \min\{C(o_1), C(o_2)\}}{\max\{C(o_1), C(o_2)\}}.$$

It should be noted that the USM approximated by Krasnogor and Pelta, $UCD(o_1, o_2)$, $NCD(o_1, o_2)$, and $ACD(o_1, o_2)$ become the same value if $C(o_1 \cdot o_2) = C(o_2 \cdot o_1)$, and both of UCD and NCD are able to give better classification accuracy than CD [20].

2.2.1 Image Compression Algorithms

We employed the following image compression algorithms.

(1) JPEG

JPEG is usually lossy compression. An image is split into blocks of eight by eight pixels. A two-dimensional forward discrete cosine transform (DCT) is calculated for all blocks. After quantization, the image is compressed using Huffman coding [30].

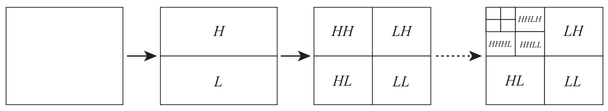


Fig. 2 Illustration on the construction of a multiresolution pyramid structure by S (Sequential) transform. S transform is alternately applied to the columns and rows of an image. The region of "*HH*", where high values are concentrated, is transformed by S transform repeatedly. Note that S transform is similar to the Haar wavelet transform.

(2) GIF

GIF is based on the Lempel-Ziv algorithm [31], which is a dictionary coder. It reads an input sequence, constructs a dictionary dynamically, and replaces the sequence with words of the dictionary.

(3) PNG

PNG is also based on the Lempel-Ziv algorithm [31], and uses Huffman coding, where PNG has been developed to replace GIF. The compression rate of PNG is often higher than that of GIF.

(4) IFS

IFS stands for Iterated Function Systems, is a quadtree-based fractal image coder/decoder, and was implemented by Polvere and Nappi [32]. The software called Mars is available from http://inls.ucsd.edu/~fisher/Fractals/Mars-1.0.tar.gz. Note that the software can accept only grayscale images using one byte memory for a pixel as raw image files.

(5) SPC

SPC is a lossless image compression, and was developed by Said and Pearlman [33]. It uses a simple pyramid multiresolution scheme enhanced with predictive coding, and contains S (Sequential) transform, which is similar to the Haar wavelet transform, and P (Prediction).

In S transform, a sequence $T[i]$ is transformed to two sequences $H[i]$ and $L[i]$ with half the length so that the average variance of the two sequences is smaller than the variance of the original sequence if the correlation coefficient of $T[2i]$ and $T[2i + 1]$ is larger than $\frac{1}{3}$. To be more precise, $L[i] = T[2i] - T[2i + 1]$ and $H[i] = \lfloor (T[2i] + T[2i + 1])/2 \rfloor$. Note that in the Haar wavelet transform, $L[i] = (T[2i] - T[2i + 1])/2$ and $H[i] = (T[2i] + T[2i + 1])/2$, and $L[i]$ and $H[i]$ can be non-integer values although in the S transform they are always integers. Since $L[i]$ often has small variance in image compression, we can reduce the errors of linearly predicted values for $L[i]$ using $H[i]$s and $L[i + 1]$. It should be noted that the distances between a residue and two consecutive residues for protein P_k, $M_k[i, j]$ and $M_k[i, j+1]$, are often almost the same, and thus $L[i]$ also has small variance as well as the case of usual images.

These transformations are done alternately to the columns and rows of an image, and the obtained region of "*HH*" is transformed repeatedly (See Fig. 2). It is expected that two-dimensional relationships that are not obtained by only sequence compression are obtained by transforming the columns and rows alternately and repeatedly, as well as other image compression algorithms.

Finally, the sequence that is obtained from the transformed image by simple raster scanning is encoded using arithmetic [34] or Huffman coding.

The software is available from http://www.cipr.rpi.edu/research/SPIHT/EW_Code/lossless.tar.gz. Note that the software can accept only grayscale images as raw image files.

2.3 Modification of SPC (MSPC)

In order to apply image compression algorithms, images must have rectangular shapes. However, the concatenated image cannot be of a rectangular shape if the lengths of two protein sequences are different. To avoid such cases, we filled the space with black color.

In addition to a simple filling method, we propose a modified procedure of SPC, called MSPC, because SPC often gave better results among the other image compressions, JPEG, GIF, PNG, and IFS, in the experiments, and SPC is simple and easy to be modified. SPC contains two parts of procedures, S+P transform for two-dimensional data and sequence encoding by arithmetic or Huffman coding (See the previous section). In order to obtain the compression size of the combined image of two images o_1 and o_2 corresponding to two proteins, $C(o_1 \cdot o_2)$, MSPC transforms o_1 and o_2 individually by S+P transform, where o_1 and o_2 are not filled with black color, and are directly calculated from the distance matrices M_1 and M_2, respectively. It should be noted that S transform is applied the same number of times for both of o_1 and o_2. After these transforms are done alternately to the columns and rows of each image, the two transformed images are transformed into two sequences by simple raster scanning, respectively. Then, the sequence for o_1 is concatenated ahead of that for o_2, and finally the concatenated sequence is encoded by arithmetic or Huffman coding (See Fig. 3). Although MSPC is applied to two proteins for measuring the similarity in this paper, we can apply it to more than two proteins for the purpose of multiple alignments of protein structures.

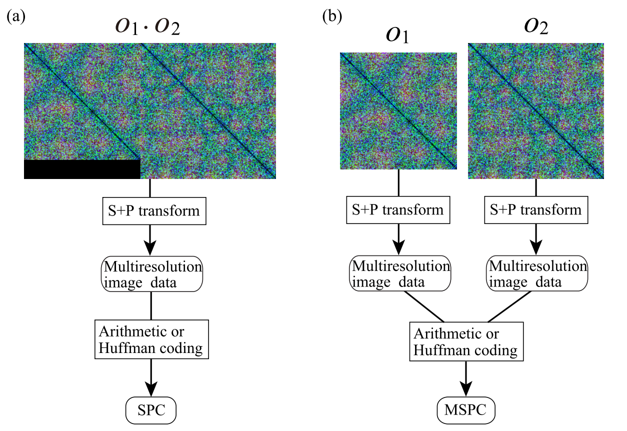

(a)
$o_1 \cdot o_2$
S+P transform
Multiresolution image data
Arithmetic or Huffman coding
SPC
(b)
o_1
o_2
S+P transform
S+P transform
Multiresolution image data
Multiresolution image data
Arithmetic or Huffman coding
MSPC

Fig. 3 Comparison of compression procedures of SPC and MSPC. Compression procedures for obtaining the compression size of the concatenated image $C(o_1 \cdot o_2)$ from two images o_1 and o_2 corresponding to two proteins. (a) Procedure of SPC. The concatenated image $o_1 \cdot o_2$ is transformed by S+P transform. (b) Procedure of MSPC. The images o_1 and o_2 are individually transformed to multiresolution image data by S+P transform, and are transformed into two sequences by simple raster scanning, respectively. Then, the transformed sequences are concatenated and encoded by arithmetic or Huffman coding. Note that S transform is applied the same number of times for both of o_1 and o_2.

Table 1 The Chew-Kedem dataset. It contains 19 mainly alpha proteins, 7 mainly beta proteins, and 10 alpha-beta proteins. Each superfamily of CATH is as follows. [a] Globins, [b] EF-hand, [c] Trp Operon Repressor; Chain A, [d] Immunoglobulins, [e] Enolase-like, N-terminal domain, [f] P-loop containing nucleotide triphosphate hydrolases, [g] Glutamine Phosphoribosylpyrophosphate, subunit 1, domain 1, [h] Divalent-metal-dependent TIM barrel enzymes.

Class of CATH	Superfamily	Proteins
Mainly alpha	1.10.490.10[a]	1ash, 1babA, 1babB, 1eca, 1flp, 1hlb, 1hlm, 1ithA, 1lh2, 1mba, 1myt, 2hbg, 2lhb, 2vhb, 2vhbA, 3sdhA, 5mbn
	others	1cnpA (1.10.238.10[b]), 1jhgA (1.10.1270.10[c])
Mainly beta	2.60.40.10[d]	1cd8, 1cdb, 1ci5A, 1hnf01, 1neu, 1qa9A, 1qfoA
Alpha-beta	3.30.390.10[e]	1chrA1, 2mnr01, 4enl01
	3.40.50.300[f]	1aa9, 1gnp, 1qraA, 5p21, 6q21A
	others	1ct9A1 (3.60.20.10[g]), 6xia (3.20.20.150[h])

3. Results on Computational Experiments

3.1 Data

We used two datasets, the Chew-Kedem dataset [35] and the Sierk-Pearson dataset [36], which were also used in [13], [20], [24]. The Chew-Kedem dataset contains proteins and domains identified by their PDB codes. We obtained their PDB-style files with coordinates from the PDB database as follows (See Table 1): 17 globins (1ash, 1babA, 1babB, 1eca, 1flp, 1hlb, 1hlm, 1ithA, 1lh2, 1mba, 1myt, 2hbg, 2lhb, 2vhb, 2vhbA, 3sdhA, 5mbn), 2 mainly alpha proteins except globins (1cnpA, 1jhgA), 7 mainly beta (immunoglobulin) proteins (1cd8, 1cdb, 1ci5A, 1hnf01, 1neu, 1qa9A, 1qfoA), 3 enolase-like, N-terminal domains (1chrA1, 2mnr01, 4enl01), 4 P-loop containing nucleotide triphosphate hydrolases (1aa9, 1gnp, 1qraA, 5p21, 6q21A) and 2 other mixed alpha-beta proteins (1ct9A1, 6xia). The Sierk-Pearson dataset contains 86 proteins and domains (20 mainly alpha proteins, 26 mainly beta proteins, and 40 mixed alpha-beta proteins) (See Table 2). We used the archive, SP-86-ATOM.tar.gz, provided by Ferragina et al. on their supplementary material web page [20], which consists of ATOM lines in the PDB entry for each protein domain.

3.2 Experiments

For each pair of proteins included in the Chew-Kedem dataset and that in the Sierk-Pearson dataset, we generated two raw image files, o_1 and o_2, and the two concatenated image files, $o_1 \cdot o_2$ and the reverse, $o_2 \cdot o_1$, from the two three-

Table 2 The Sierk-Pearson dataset. It contains 20 mainly alpha proteins, 26 mainly beta proteins, and 40 alpha-beta proteins.

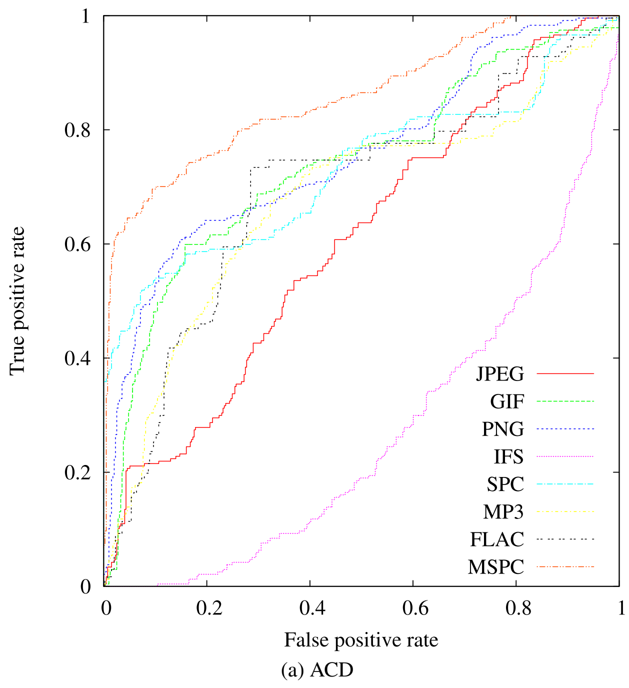
Class of CATH	Proteins
Mainly alpha	1ad6A, 1ao6A5, 1bbhA0, 1cnsA1, 1d2zD0, 1dat00, 1e12A0, 1eqzE0, 1gwxA0, 1hgu00, 1hlm00, 1jnk02, 1mmoD0, 1nubA0, 1quuA1, 1repC1, 1sw6A0, 1trrA0, 2hpdA0, 2mtaC0
Mainly beta	1a8d02, 1a8h02, 1aozA3, 1b8mB0, 1bf203, 1bjqB0, 1bqyA2, 1btkB0, 1c1zA5, 1cl7H0, 1d3sA0,1danU0, 1dsyA0, 1dxmA0, 1et6A2, 1extB1, 1nfiC1, 1nukA0, 1otcA1, 1qdmA2, 1qe6D0, 1qfkL2, 1que01, 1rmg00, 1tmo04, 2tbvC0
Alpha-beta	1a1mA1, 1a2vA2, 1akn00, 1aqzB0, 1asyA2, 1atiA2, 1auq00, 1ax4A1, 1b0pA6, 1b2rA2, 1bcg00, 1bcmA1, 1bf5A4, 1bkcE0, 1bp7A0, 1c4kA2, 1cd2A0, 1cdg01, 1d0nA4, 1d4oA0, 1d7oA0, 1doi00, 1dy0A0, 1e2kB0, 1eccA1, 1fbnA0, 1gsoA3, 1mpyA2, 1obr00, 1p3801, 1pty00, 1qb7A0, 1qmvA0, 1urnA0, 1zfjA0, 2acy00, 2drpA1, 2nmtA2, 2reb01, 4mdhA2

dimensional structures. For MSPC, the compression size is calculated not from a concatenated image but from a pair of images. We applied the above compression algorithms, JPEG, GIF, PNG, IFS, SPC, and MSPC, respectively, to the raw files, and calculated $ACD(o_1, o_2)$ and $NCD(o_1, o_2)$. We obtained hierarchical clustering results using the single linkage, average linkage, and Ward method [37].

These experiments were done in a single processor core on a PC with Xeon X5460 3.16 GHz CPUs and 8 GB memory under the Linux (version 2.6) operating system, where the gcc compiler was used with optimization option -O2. The source codes implemented in this paper and the results on the similarity matrices are available on our supplementary information web page http://sunflower.kuicr.kyoto-u.ac.jp/morihiro/imgcomp/. The average compression elapsed time by MSPC that obtains $C(o_1)$, $C(o_2)$, and $C(o_1 \cdot o_2)$ for each protein pair of the Sierk-Pearson dataset was 0.030 seconds. It is suggested that compression-based methods including MSPC are very efficient.

3.2.1 ROC Curves and F-Measures

In order to evaluate the similarity measure, we used Receiver Operating Characteristic (ROC) analysis [38] as in [20], [24]. We considered the binary classification problem whether a pair of proteins is contained in the same CATH class or not. For the dataset having N proteins, each of $\binom{N}{2}$ pairs is classified in either class 'T' (the pair is in the same CATH class) or 'F' (the pair is not in the same CATH class). An ROC curve is plotted using the sensitivity (the true positive rate, $TP/(TP + FN)$) and one minus the specificity (the false positive rate, $FP/(TN + FP)$) for a binary classification predictor as the threshold changes, where TP, FP, TN, and FN are the numbers of true positives, false positives, true negatives, false negatives, respectively. The area under the curve (AUC) is often used as a measure of overall classification accuracy. The AUC for a random classifier takes the value around 0.5, and then the ROC curve lies along the diagonal. The closer the ROC curve comes to the upper left point, (0, 1), the better the classification accuracy is. The AUC for a perfect classification is 1.



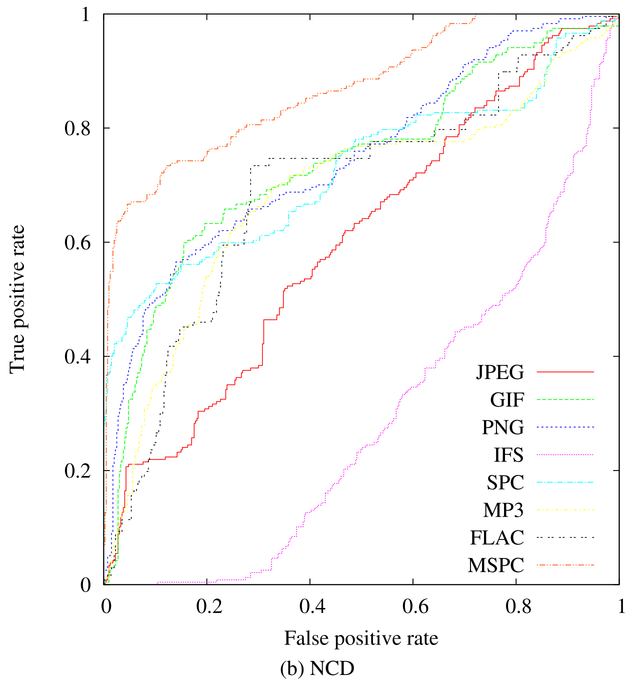
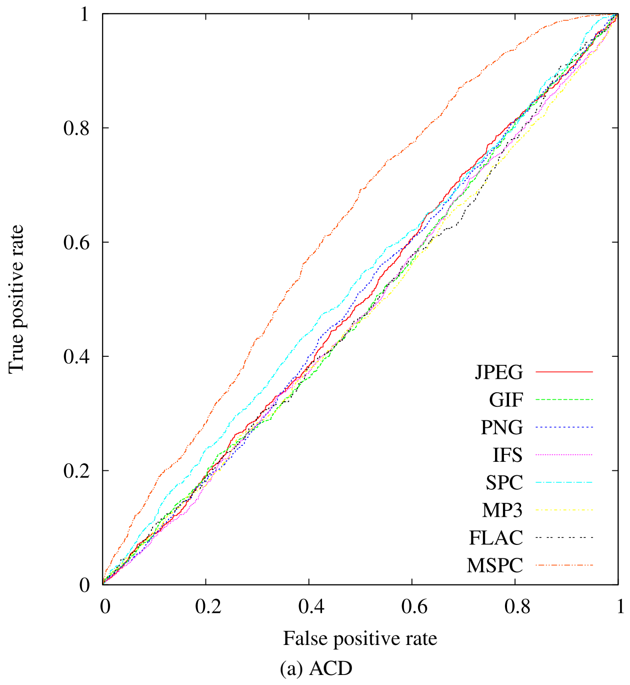


Fig. 4 ROC curves for the Chew-Kedem dataset measured by (a) ACD and (b) NCD using image compressions, JPEG, GIF, PNG, IFS, SPC, and MSPC.

As another evaluation method, we used F-measure which was used also in [20]. We employed their program, f-measure.pl, that is available on their supplementary material web page. F-measure is defined to be the harmonic mean of the precision ($P = TP/(TP + FP)$) and the recall (the true positive rate, $R = TP/(TP + FN)$) as $\frac{2PR}{P+R}$, where the bast value is 1.

Figures 4 and 5 show ROC curves for the Chew-Kedem



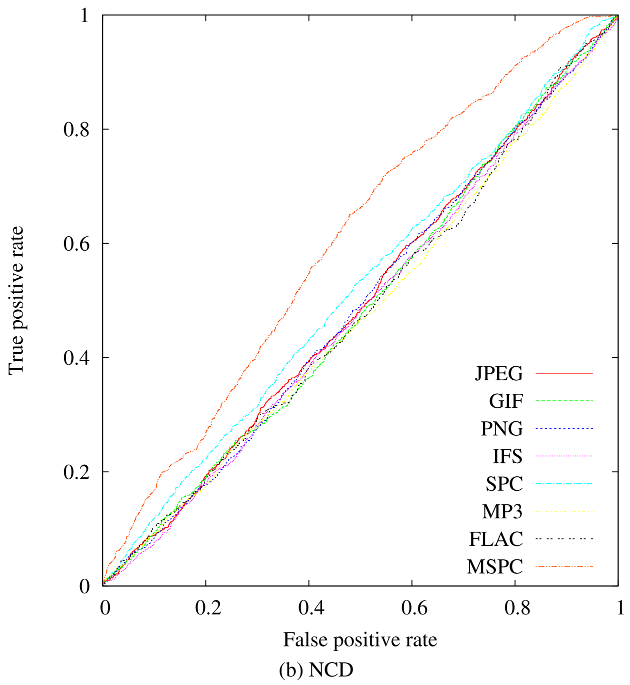


Fig. 5 ROC curves for the Sierk-Pearson dataset measured by (a) ACD and (b) NCD using image compressions, JPEG, GIF, PNG, IFS, SPC, and MSPC.

Table 3 Results on AUC and F-measure for the Chew-Kedem dataset measured by ACD and NCD using image compressions, JPEG, GIF, PNG, IFS, SPC, and MSPC.

Compression algorithm	AUC by ACD	F-measure by ACD Single	Average	Ward
JPEG	0.6051	0.6502	0.6722	0.7660
GIF	0.7365	0.7440	0.8210	0.8210
PNG	0.7545	0.7947	0.7930	0.7613
IFS	0.2674	0.5875	0.5737	0.6114
SPC	0.7296	0.8274	0.8367	0.8838
MSPC	0.8565	0.8211	0.8516	**0.9445**
Compression algorithm	AUC by NCD	F-measure by NCD Single	Average	Ward
JPEG	0.6020	0.6641	0.6722	0.7660
GIF	0.7381	0.7440	0.8210	0.8210
PNG	0.7469	0.7947	0.7930	0.7613
IFS	0.2885	0.5929	0.5767	0.6114
SPC	0.7267	0.8274	0.8367	0.8838
MSPC	**0.8687**	0.8211	**0.8781**	**0.9445**
Krasnogor and Pelta [13]	0.7957	**0.8379**	0.8379	0.8379
Dai and Wang [24]	0.86	-	-	-

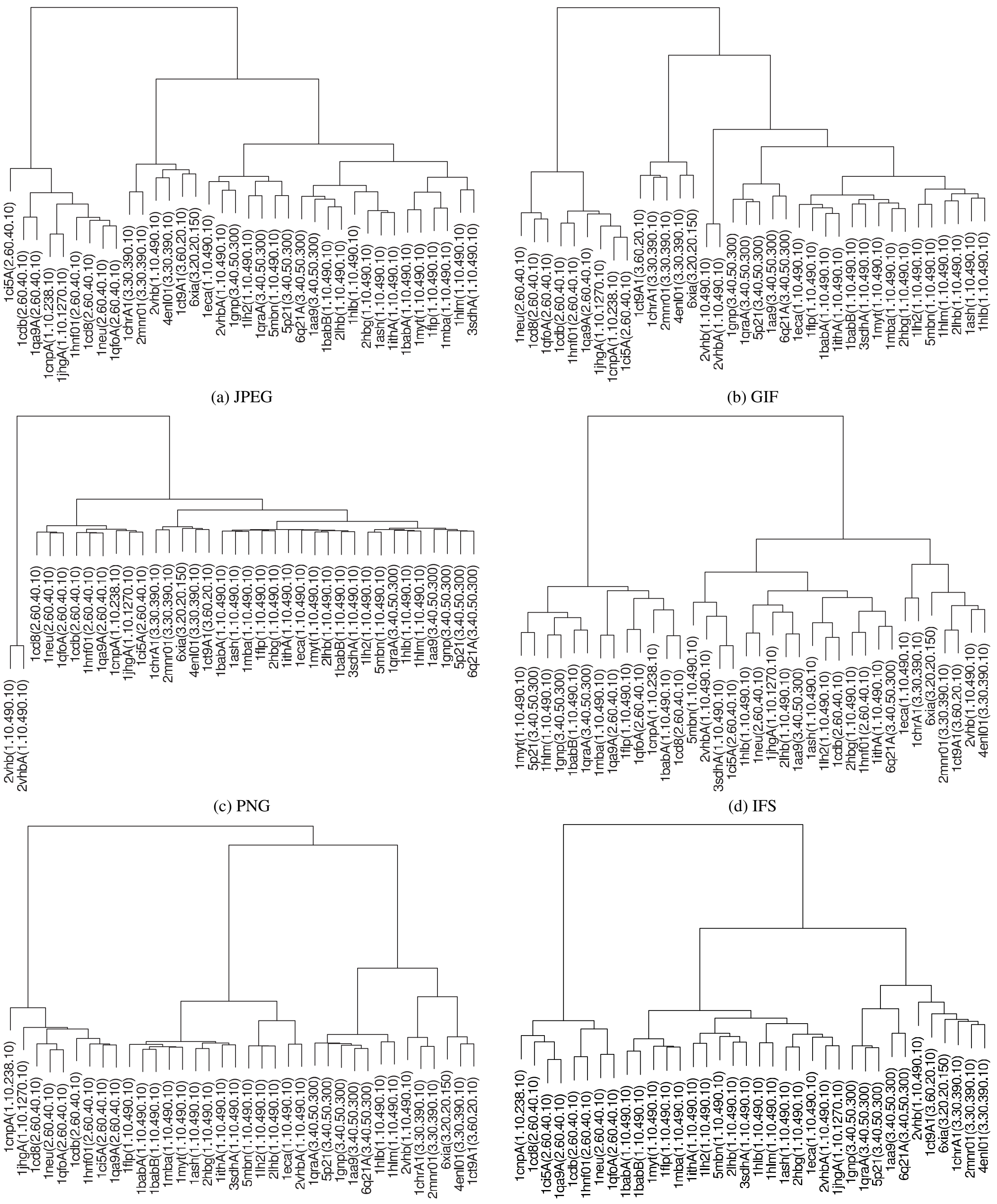
Table 4 Results on AUC and F-measure for the Sierk-Pearson dataset measured by ACD and NCD using image compressions, JPEG, GIF, PNG, IFS, SPC, and MSPC. The result by Ferragina et al. [a] is that of the compression algorithm, 'BwtMtfRleRc fast', with the similarity measure, UCD and NCD, where the F-measure was the best for SP-86-ATOM [20].

Compression algorithm	AUC by ACD	F-measure by ACD Single	Average	Ward
JPEG	0.4988	0.5485	0.5387	0.5234
GIF	0.4866	0.5454	0.5455	0.5303
PNG	0.4978	0.5414	0.5299	0.5248
IFS	0.4807	0.5540	0.5559	0.5390
SPC	0.5227	0.5352	0.5251	0.5407
MSPC	**0.6228**	**0.5724**	**0.6197**	**0.6056**
Compression algorithm	AUC by NCD	F-measure by NCD Single	Average	Ward
JPEG	0.4949	0.5473	0.5378	0.5265
GIF	0.4870	0.5513	0.5386	0.5316
PNG	0.4906	0.5402	0.5366	0.5248
IFS	0.4827	0.5428	0.5599	0.5234
SPC	0.5195	0.5277	0.5335	0.5265
MSPC	0.6052	0.5621	0.5966	0.6046
Krasnogor and Pelta [13]	0.5033	0.5581	0.5377	0.5497
Ferragina et al. [20]	-	-	0.5791[a]	-
Dai and Wang [24]	0.575	-	-	-

and Sierk-Pearson datasets measured by ACD and NCD using image compressions, JPEG, GIF, PNG, IFS, SPC, and MSPC, respectively. We can see from the figures that the results using MSPC were better than those using other compression algorithms, JPEG, GIF, PNG, IFS, and SPC for both datasets and for both similarity measures. It is suggested that our modification to the concatenation of two images works well for measuring the similarity of protein structures.

Tables 3 and 4 show the results on AUC and F-measure for the Chew-Kedem and Sierk-Pearson datasets measured by ACD and NCD using image compressions, JPEG, GIF, PNG, IFS, SPC, and MSPC, respectively. For the Chew-Kedem dataset, we can see from the table that the AUC for MSPC measured by NCD was the best, was better than that by Krasnogor and Pelta, where their method is based on the compression of contact maps from protein structures, and was comparable with that by Dai and Wang, where their method is based on the compression of protein sequences. In the F-measures, the Ward method gave the best score when we used MSPC as the compression method. For the Sierk-Pearson dataset, the AUC for MSPC measured by ACD was the best, and was better than those by Krasnogor and Pelta, and Dai and Wang. The F-measures for the result using the average linkage method (UPGMA), MSPC and ACD, was the best, and was better than those by Krasnogor and Pelta, and Ferragina et al. It should be noted that the Chew-Kedem dataset is high redundant with respect to protein sequences, the Sierk-Pearson dataset is less redundant [24], and the combination of MSPC and ACD was able to measure the similarity of protein structures with less redundant sequences better than the other method.

Figure 6 shows the clustering results using the Ward



(a) JPEG (b) GIF (c) PNG (d) IFS (e) SPC (f) MSPC

Fig. 6 Clustering results using the Ward method for the Chew-Kedem dataset measured by ACD. Applied image compression algorithms are (a) JPEG, (b) GIF, (c) PNG, (d) IFS, (e) SPC, and (f) MSPC, respectively. The number in parenthesis denotes the superfamily of CATH.

method for the Chew-Kedem dataset measured by ACD for the compression algorithms, JPEG, GIF, PNG, IFS, SPC, and MSPC. We can see from the figures that MSPC classified the dataset best as shown in the previous results. MSPC misclassified only two mainly alpha proteins, 1cnpA was classified in mainly beta class, and 2vhb was in alpha-beta class. However, MSPC recognized that the cluster of mainly alpha proteins was more similar to that of alpha-beta proteins than that of mainly beta proteins, and classified 3 proteins in the superfamily 3.30.390.10 and 5 proteins in 3.40.50.300 correctly (See Table 1).

4. Theoretical Analysis

In the previous sections, we proposed the methods for measuring the similarity of protein structures. In this section, we consider the problem of comparing protein tertiary structures by image comparison from a view point of alignment of distance matrices [6], and analyze the time complexity. It should be noted that problems we define here are very similar to those of the contact map overlap (CMO), and results concerning the time complexity are derived by using techniques similar to those for CMO [39]–[41].

Next, we derive a relationship between our defined problem and Kolmogorov complexity. Furthermore, we qualitatively explain using the entropy the reason why MSPC achieves higher compression ratios than SPC.

4.1 Time Complexity

In the computational experiments, we used the Chew-Kedem and Sierk-Pearson datasets. The compression ratio of GIF and PNG for the raw image files was more than 0.408, and that of JPEG, usually used as lossy compression, was more than 0.283. Therefore, in practice, distance matrices with size $n \times n$ can be regarded to be compressed to $\Theta(n^2 \log m)$ bits, where m denotes the maximum of pixel values and we assume that each distance is truncated into $O(\log m)$ bits. On the other hand, a distance matrix is theoretically calculated from n three-dimensional coordinates and thus the matrix might be compressed into $O(n \log m)$ bits. For example, if $n = 50$, that is, a protein contains 50 amino acids, and the distance matrix with size 50×50 is compressed to $O(n \log m)$ bits, then the compression ratio would be about $1/50 = 0.02$. However, the above experimental results suggest that the sizes of compressed images are still $\Theta(n^2 \log m)$ bits in practice. Therefore, in the following, we will assume that distance matrices are compressed to $\Theta(n^2 \log m)$ bits.

In order to analyze similarities between such distance matrices, we consider an alignment $\sigma = \{(i_1, j_1), \cdots, (i_k, j_k)\}$ $(1 \le i_1 < \cdots < i_k \le n_1, 1 \le j_1, \cdots, j_k \le n_2)$ between the distance matrices M_1 with size $n_1 \times n_1$ and M_2 with size $n_2 \times n_2$, where position i_l $(l = 1, \cdots, k)$ in a protein corresponds to j_l in the other protein and σ can be considered as the function defined by the domain $D(\sigma) = \{i_1, \cdots, i_k\}$ and $\sigma(i_l) = j_l$ $(l = 1, \cdots, k)$. Then, we consider the following problem, where we use scoring functions simpler than that in [6] for the purpose of theoretical studies.

Problem 1: Given two distance matrices $M_1 = (x_{ij})$, $M_2 = (y_{ij})$, and a positive integer k $(\le \min\{n_1, n_2\})$, find an alignment σ with size $|D(\sigma)| = k$ that minimizes

$$f_1(M_1, M_2, \sigma) = \sum_{i \in D(\sigma)} \sum_{j \in D(\sigma)} (x_{ij} - y_{\sigma(i)\sigma(j)})^2. \tag{1}$$

Theorem 1: Problem 1 is NP-hard.

Proof. We show that there exists a polynomial-time reduction from the maximum clique (Max-Clique) problem. Max-Clique is defined as follows: Given an undirected graph $G(V, E)$ with a set of vertices V and a set of edges E and a positive integer k, determine whether or not there is a k-clique in G. We define two matrices $M_1 = (x_{ij})$ with size $k \times k$ and $M_2 = (y_{ij})$ with size $|V| \times |V|$ from an instance of Max-Clique as follows.

$$x_{ij} = \begin{cases} 0 & (i = j) \\ 1 & (i \ne j) \end{cases} \tag{2}$$

$$y_{ij} = \begin{cases} 0 & (i = j \text{ or } (i, j) \notin E) \\ 1 & ((i, j) \in E) \end{cases} \tag{3}$$

Then, there exists a k-clique if and only if there exists σ such that $f_1(M_1, M_2, \sigma) = 0$. Furthermore, a k-clique can be constructed from such σ in polynomial time. Therefore, there exists a polynomial-time reduction from Max-Clique, which is NP-complete. □

In addition, Problem 1 has no polynomial-time algorithm with a guaranteed approximation ratio unless P=NP because the minimum value can be 0.

We can also consider the problem of maximizing the number of aligned pairs under the condition of $f_1(M_1, M_2, \sigma) = 0$ and the problem of maximizing the number of matched pixels.

Problem 2: Given two distance matrices $M_1 = (x_{ij})$, $M_2 = (y_{ij})$, find an alignment σ with maximum k such that $f_1(M_1, M_2, \sigma) = 0$.

Problem 3: Given two distance matrices $M_1 = (x_{ij})$, $M_2 = (y_{ij})$, and a positive integer k, find an alignment σ with size $|D(\sigma)| = k$ that maximizes

$$f_2(M_1, M_2, \sigma) = \sum_{i \in D(\sigma)} \sum_{j \in D(\sigma)} \delta(x_{ij}, y_{\sigma(i)\sigma(j)}), \tag{4}$$

where $\delta(x, y) = 1$ if $x = y$, otherwise 0.

Theorem 2: Both Problem 2 and Problem 3 are NP-hard.

Proof. We use the same reduction as in the proof of Theorem 1. Then, it is straight-forward to see that it is a polynomial-time reduction for both problems. □

Unlike Problem 1, there exists an approximation algorithm for Problem 3.

Theorem 3: Problem 3 can be approximated within a factor of $O(n)$ in polynomial time.

Proof. For each i_0, i'_0 $(1 \leq i_0 \leq n_1, 1 \leq i'_0 \leq n_2)$, we find an optimal alignment σ_{i_0,i'_0} between two sequences $x_{i_0 j}$ $(1 \leq j \leq n_1)$ and $y_{i'_0 j'}$ $(1 \leq j' \leq n_2)$ that maps $x_{i_0 i_0}$ to $y_{i'_0 i'_0}$ using dynamic programming as follows. For $i < i_0$, $i' < i'_0$,

$$D[i, i'] = \max \begin{cases} D[i-1, i'-1] + \delta(x_{i_0,i}, y_{i'_0,i'}) \\ D[i-1, i'] \\ D[i, i'-1] \end{cases}, \quad (5)$$

and for $i > i_0$, $i' > i'_0$,

$$D[i, i'] = \max \begin{cases} D[i+1, i'+1] + \delta(x_{i_0,i}, y_{i'_0,i'}) \\ D[i+1, i'] \\ D[i, i'+1] \end{cases}, \quad (6)$$

where the initialization of the matrix is straight-forward. Let s_{i_0,i'_0} be the score of the alignment σ_{i_0,i'_0}, $\sum_{j \in D(\sigma_{i_0,i'_0})} \delta(x_{i_0 j}, y_{i'_0 \sigma_{i_0,i'_0}(j)})$. Then, the algorithm outputs the alignment σ_{i_0,i'_0} that maximizes s_{i_0,i'_0}. The maximum of s_{i_0,i'_0} is at least $1/n$ of the maximum of $f_2(M_1, M_2, \sigma)$, and the algorithm runs in polynomial time. □

It is to be noted that this $O(n)$ approximation ratio is meaningful because the size of the input is $\Theta(n^2 \log m)$ bits.

4.2 Kolmogorov Complexity

Suppose that an optimal alignment σ with size k such that $f_1(M_1, M_2, \sigma) = 0$ is obtained for the matrices M_1 and M_2 by solving Problem 2. Then, we can construct M_2 from M_1 as follows. First, we delete d_1 $(= n_1 - k)$ rows and d_1 columns from M_1 that are not included in $D(\sigma)$. Next, we add d_2 $(= n_2 - k)$ rows and d_2 columns. This procedure requires $O(d_1 \log n)$ bits for the specification of rows and columns to be deleted, and $O(d_2 n \log n \log m)$ bits for the addition of d_2 columns and rows. Therefore, we have the following for the Kolmogorov complexities $K(M_1)$ and $K(M_2)$.

Theorem 4: For distance matrices M_1 and M_2,

$$K(M_2) \leq K(M_1) + O(d_2 n \log n \log m) + O(d_1 \log n),$$
$$K(M_1) \leq K(M_2) + O(d_1 n \log n \log m) + O(d_2 \log n).$$

Since it can be assumed from the previous subsection that distance matrices are compressed to $\Theta(n^2 \log m)$ bits in practice, it means that the Kolmogorov complexity of M_1 is not so large compared with that of M_2 if the distance (d_1+d_2) between M_1 and M_2 is small.

Let $M_3 = (z_{ij})$ be the concatenated image of $M_1 = (x_{ij})$ with size $n_1 \times n_1$ and M_2 with size $n_2 \times n_2$ $(n_1 < n_2)$, where $z_{ij} = x_{ij}$ for $i, j \leq n_1$. Then, the size of M_3 is $n_2 \times (n_1 + n_2)$, and $(n_2 - n_1) \times n_1$ is the additional black-colored region. Since $O(\log n)$ bits are needed for specifying the black region, $K(M_3) \leq K(M_1) + K(M_2) + O(\log n)$. This gives an upper bound of the compressed size of concatenated images.

The S transform in SPC tries to reduce the entropy of images. We can assume that $|x_{ij} - x_{i(j+1)}|$ for $j < n_1$ is relatively small because the distance between the C_α atom of jth residue and that of $(j+1)$th residue is small. Then, it should be noted that the entropy of the region indicated by 'L' in Fig. 2 is also small for individual images of M_1 and M_2. However, for the concatenated M_3, since $z_{i(n_1+1)} = 0$, $|z_{in_1} - z_{i(n_1+1)}| = x_{in_1}$, and the value will not be vertically transformed by the S transform again because it is in the L region. Although the value is horizontally transformed by the S transform, it is considered that the entropy of the transformed HL region (see Fig. 2) is still high. Thus, since MSPC does not handle such a black region, the compression size of the concatenated image by SPC can be larger than that by MSPC if the concatenated image includes the black region. In fact, in the computational experiments, the ratio of protein pairs that the compression size by SPC was larger than or equal to that by MSPC was 0.993. It suggests that MSPC approximates the Kolmogorov complexity more accurately than SPC in most cases, and MSPC can estimate the similarity of protein structures more accurately than SPC.

5. Conclusions

We proposed an image compression-based approach to measuring the similarity of protein structures, and applied them to the Chew-Kedem dataset and the Sierk-Pearson dataset. The results on ROC analysis and F-measure for our proposed method MSPC, which is obtained by modifying SPC image compression algorithm, were the best among several image compression algorithms, were better than that by Krasnogor and Pelta, and were comparable to or better than that by Dai and Wang for the Chew-Kedem dataset. Although the results for the Sierk-Pearson dataset were not so good for all the methods, those for MSPC were better than those by Krasnogor and Pelta, Ferragina et al. and Dai and Wang where Krasnogor and Pelta, and Ferragina et al. use protein structure data, and Dai and Wang uses only sequence data. Moreover, the result on the elapsed time showed that MSPC is very efficient.

Almost all image compression algorithms have been developed based on the property that neighbor pixels often have similar colors in images. Unlike sequence compression, image compression algorithms compress data horizontally and vertically. However, it is considered from the clustering result that MSPC does not always compresses similar substructures located at distant locations sufficiently. Therefore, it is expected that better similarity measure can be obtained by improving some image compression algorithm as we modified SPC and obtained MSPC in this paper. In addition, values handled by image compression algorithms are restricted to integers of a few bytes. In this paper, we transformed distances between residues of a protein to integers. In future work, we would like to develop compression algorithms for distances with real values.

Acknowledgments

This work was partially supported by Grants-in-Aid "Systems Genomics", #19650053 and #21700323 from MEXT, Japan.

References

[1] A. Andreeva, D. Howorth, S.E. Brenner, T.J.P. Hubbard, C. Chothia, and A.G. Murzin, "SCOP database in 2004: Refinements integrate structure and sequence family data," Nucl. Acids Res., vol.32, pp.D226–D229, 2004.

[2] F.M. Pearl, C.F. Bennett, J.E. Bray, A.P. Harrison, N. Martin, A. Shepherd, I. Sillitoe, J. Thornton, and C.A. Orengo, "The CATH database: An extended protein family resource for structural and functional genomics," Nucl. Acids Res., vol.31, pp.452–455, 2003.

[3] W.R. Taylor and C.A. Orengo, "Protein structure alignment," J. Molecular Biology, vol.208, pp.1–22, 1989.

[4] T. Akutsu, "Protein structure alignment using dynamic programming and iterative improvement," IEICE Trans. Inf. & Syst., vol.E79-D, no.12, pp.1629–1636, Dec. 1996.

[5] I.N. Shindyalov and P.E. Bourne, "Protein structure alignment by incremental combinatorial extension (CE) of the optimal path," Protein Engineering, vol.11, pp.739–747, 1998.

[6] L. Holm and C. Sander, "Protein structure comparison by alignment of distance matrices," J. Molecular Biology, vol.233, pp.123–138, 1993.

[7] Y. Ye and A. Godzik, "Multiple flexible structure alignment using partial order graphs," Bioinformatics, vol.21, pp.2362–2369, 2005.

[8] A. Caprara, R. Carr, S. Istrail, G. Lancia, and B. Walenz, "1001 optimal PDB structure alignments: Integer programming methods for finding the maximum contact map overlap," J. Computational Biology, vol.11, pp.27–52, 2004.

[9] Y. Wang, L.Y. Wu, X.S. Zhang, and L. Chen, "Automatic classification of protein structures based on convex hull representation by integrated neural network," Lect. Notes Comput. Sci., vol.3959, pp.505–514, 2006.

[10] Y. Nonomura, K. Yoshino, T. Nakae, and T. Ohkawa, "Retrieval of protein with similar interaction based on structural data of protein-compound complex," Trans. IPSJ, vol.47, no.SIG1(TOM14), pp.110–119, 2006.

[11] Y. Kiuchi, T. Ozaki, and T. Ohkawa, "Partial geometric hashing for retrieving similar interaction protein using profile," International Conference on Information Technology (ITNG'07), pp.589–596, 2007.

[12] T. Shibuya, J. Jansson, and K. Sadakane, "Linear-time protein 3-D structure searching with insertions and deletions," Algorithms for Molecular Biology, vol.5, p.7, 2010.

[13] N. Krasnogor and D.A. Pelta, "Measuring the similarity of protein structures by means of the universal similarity metric," Bioinformatics, vol.20, pp.1015–1021, 2004.

[14] D. Barthel, J. Hirst, J. Błażewicz, E. Burke, and N. Krasnogor, "ProCKSI: A decision support system for protein (structure) comparison, knowledge, similarity and information," BMC Bioinformatics, vol.8, p.416, 2007.

[15] A. Shah, G. Folino, and N. Krasnogor, "Toward high-throughput, multicriteria protein-structure comparison and analysis," IEEE Trans. NanoBioscience, vol.9, no.2, pp.144–155, 2010.

[16] M. Li, J.H. Badger, X. Chen, S. Kwong, P.E. Kearney, and H. Zhang, "An information-based sequence distance and its application to whole mitochondrial genome phylogeny," Bioinformatics, vol.17, pp.149–154, 2001.

[17] A. Kocsor, A. Kertész-Farkas, L. Kaján, and S. Pongor, "Application of compression-based distance measures to protein sequence classification: A methodological study," Bioinformatics, vol.22, pp.407–412, 2006.

[18] D. Pelta, N. Krasnogor, C. Bousono-Calzon, J. Verdegay, J. Hirst, and E. Burke, "A fuzzy sets based generalization of contact maps for the overlap of protein structures," J. Fuzzy Sets and Systems, vol.152, pp.103–123, 2005.

[19] G. Terrazas, P. Siepmann, G. Kendall, and N. Krasnogor, "An evolutionary methodology for the automated design of cellular automaton-based complex systems," J. Cellular Automata, vol.2, pp.77–102, 2007.

[20] P. Ferragina, R. Giancarlo, V. Greco, G. Manzini, and G. Valiente, "Compression-based classification of biological sequences and structures via universal similarity metric: Experimental assessment," BMC Bioinformatics, vol.8, p.252, 2007.

[21] D. Shkarin, "PPM: One step to practicality," Proc. IEEE Data Compression Conference, pp.202–211, 2002.

[22] D. Shkarin, "PPMd compressor," 2006.

[23] X. Chen, S. Kwong, and M. Li, "A compression algorithm for DNA sequences," IEEE Eng. Med. Biol. Mag., vol.20, no.4, pp.61–66, 2001.

[24] Q. Dai and T. Wang, "Comparison study on k-word statistical measures for protein: From sequence to 'sequence space'," BMC Bioinformatics, vol.9, p.394, 2008.

[25] S. Henikoff and J.G. Henikoff, "Amino acid substitution matrices from protein blocks," Proc. Natl. Acad. Sci. USA, vol.89, pp.10915–10919, 1992.

[26] M.O. Dayhoff and R.V. Eck, Atlas of protein sequence and structure, Natl. Biomed. Res. Found., Silver Spring, MD, 1968.

[27] M. Li and P. Vitanyi, An Introduction to Kolmogorov Complexity and Its Applications, Springer, 1997.

[28] D.R. Gilbert, D.R. Westhead, N. Nagano, and J.M. Thornton, "Motif-based searching in TOPS protein topology databases," Bioinformatics, vol.15, pp.317–326, 1999.

[29] K. Henrick, Z. Feng, W.F. Bluhm, D. Dimitropoulos, J.F. Doreleijers, S. Dutta, J.L. Flippen-Anderson, J. Ionides, C. Kamada, E. Krissinel, C.L. Lawson, J.L. Markley, H. Nakamura, R. Newman, Y. Shimizu, J. Swaminathan, S. Velankar, J. Ory, E.L. Ulrich, W. Vranken, J. Westbrook, R. Yamashita, H. Yang, J. Young, M. Yousufuddin, and H.M. Berman, "Remediation of the protein data bank archive," Nucl. Acids Res., vol.36, pp.D426–433, 2008.

[30] D.A. Huffman, "A method for the construction of minimum-redundancy codes," Proc. Institute of Radio Engineers, pp.1098–1101, 1952.

[31] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," IEEE Trans. Inf. Theory, vol.IT-24, no.5, pp.530–536, 1978.

[32] M. Polvere and M. Nappi, "A feature vector technique for fast fractal image coding," Tech. Rep., University of Salerno, 1998.

[33] A. Said and W.A. Pearlman, "Reversible image compression via multiresolution representation and predictive coding," Proc. SPIE, pp.664–674, 1993.

[34] J.J. Rissanen, "Generalized Kraft inequality and arithmetic coding," IBM J. Res. Dev., vol.20, pp.198–203, 1976.

[35] L.P. Chew and K. Kedem, "Finding consensus shape for a protein family," Algorithmica, vol.38, pp.115–129, 2003.

[36] M.L. Sierk and W.R. Pearson, "Sensitivity and selectivity in protein structure comparison," Protein Science, vol.13, pp.773–785, 2004.

[37] J.H. Ward, "Hierarchical grouping to optimize an objective function," J. American Statistical Association, vol.58, pp.236–244, 1963.

[38] C.E. Metz, "ROC methodology in radiologic imaging," Investigative Radiology, vol.21, pp.720–733, 1986.

[39] P. Agarwal, N. Mustafa, and Y. Wang, "Fast molecular shape matching using contact maps," J. Computational Biology, vol.14, pp.131–143, 2007.

[40] T. Akutsu and S. Miyano, "On the approximation of protein threading," Theor. Comput. Sci., vol.210, pp.261–275, 1999.

[41] D. Goldman, S. Istrail, and C. Papadimitriou, "Algorithmic aspects of protein structure similarity," IEEE Symp. Found. Comput. Sci., pp.512–522, 1999.

Morihiro Hayashida received his M.Sc. degree in Information Science from University of Tokyo, Japan, and his Ph.D. degree in Informatics from Kyoto University, Japan, in 2005. He is currently an assistant professor in Laboratory of Biological Information Networks, Bioinformatics Center, Institute for Chemical Research, Kyoto University. His research interests include functional analysis of proteins and development of computational methods for bioinformatics.

Tatsuya Akutsu received his M.Eng. degree in Aeronautics in 1986 and a Dr.Eng. degree in Information Engineering in 1989 both from University of Tokyo, Japan. From 1989 to 1994, he was with Mechanical Engineering Laboratory, Japan. He was an associate professor in Gunma University from 1994 to 1996 and in Human Genome Center, University of Tokyo from 1996 to 2001 respectively. He joined Bioinformatics Center, Institute for Chemical Research, Kyoto University, Japan as a professor in Oct. 2001. His research interests include bioinformatics and discrete algorithms.