Analyzing Order of Domains in Grammar-based Compression of Proteomes

Morihiro Hayashida[1,a)] Kousei Ishibashi[1] Hitoshi Koyano[2]

Abstract: The evolutionary history of an organism has constructed DNA nucleotide sequences by natural selection. Several genes were duplicated, and nucleotide sequences have been modified by the emancipation from natural selection. From the viewpoint of compression, the identical subsequences can be replaced with the same symbol. In the previous study, a protein was regarded as a set of domains, and the proteome in an organism was compressed based on a grammar concerning sets. In this study, we regard a protein as a sequence of domains, propose an adequate grammar to compress a proteome, and compare the compression results between cases of considering with and without the order of domains in a protein for seven organisms, *Escherichia coli*, *Saccharomyces cerevisiae*, *Arabidopsis thaliana*, *Caenorhabditis elegans*, *Drosophila melanogaster*, *Mus musculus*, and *Homo sapiens*.

1. Introduction

Abundant data of DNA and protein sequences are available due to recent biological sequencing technology. In order to store, to do processing, and to analyze mass sequencing data, many compression methods have been developed. For DNA sequences, biocompress-2 compresses the sequence by detecting regularities including the presence of palindromes [1]. Cfact algorithm makes use of suffix trees, and detects the longest exact matching repeat [2]. GenCompress finds approximate matches satisfying the condition that the length of sequence of edit operations, insertion, deletion, and substitution, is less than a threshold [3]. CTW+LZ combines CTW (Context Tree Weighting) method [4] and LZ (Lempel Ziv) algorithm [5] to encode long repeating subsequences [6]. DNACompress finds approximate repeats including complemented palindromes using PatternHunter [7], which achieved better compression ratios than GenCompress and CTW+LZ [8]. Expert model (XM) was proposed to estimate the probability distribution of the next symbol in the sequence. Their method compressed better than biocompress-2, GenCompress, DNACompress, DNAPack [9], CDNA [10], and GeMNL [11] for most DNA sequences, and better than CP [12] and CTW+LZ, and slightly better than ProtComp [13] for protein sequences [14].

Hosseini et al. comprehensively compared existing compression approaches for biological data in different file formats, and reported that MFCompress [15] outperformed DELIMINATE [16], gzip [17], and LZMA [18] in terms of compression ratio for genomic sequences in multi-FASTA, and that SCALCE [19] outperformed Fqzcomp [20], Quip [21], DSRC [22], gzip, and LZMA for FASTQ sequences, which include the quality scores [23]. CaBLASTP achieved a faster speed than BLAST by searching in the compressed database [24]. CAD uses a changing dictionary of actively used amino acid residues in addition to Huffman coding, and achieved better compression rates than existing compression algorithms [25].

Another approach to compression of a proteome was proposed [26], where amino acid sequences were not directly compressed. From an evolutionary point of view, they focused on the process that proteins have obtained functions and domains, regarded a protein as a set of domains, and compressed a set of sets of domains. It was reported that the ratio of the compressed size to the original size was smaller in higher organisms such as *Homo sapiens* and *Mus musculus*, and the same domain would be frequently utilized in higher organisms.

In this study, we regard a protein as a sequence of domains to analyze the order of domains in a protein, and propose a grammar-based compression method for a set of sequences of domains. Then, we compare the compression results between cases of considering with and without the order of domains in a protein for seven organisms, *Escherichiai coli*, *Saccharomyces cerevisiae*, *Arabidopsis thaliana*, *Caenorhabditis elegans*, *Drosophila melanogaster*, *Mus musculus*, and *Homo sapiens*.

2. Methods

In this section, we briefly review the previous compression method for a set of sets of domains [26], and propose our compression method by extending the previous method.

2.1 Compression with Domains Unordered

In a general way, all identical data are replaced with the same

[1] Department of Electrical Engineering and Computer Science, National Institute of Technology, Matsue College, Matsue, Shimane 690-8518, Japan
[2] School of Life Science and Technology, Tokyo Institute of Technology, Meguro, Tokyo 152-8550, Japan
[a)] morihiro@matsue-ct.jp

symbol during compression processes. If a protein has the same subset of domains as another protein, the subset should be replaced with the same symbol. On the other hand, in the evolutionary process, genetic sequences have been copied from another part of chromosome, and the duplicated sequences can be compressed. Following the evolutionary model by Nacher et al. [27], mutation, gene duplication and fusion events were introduced into their grammar for sets of domains, which consists of three types of production rules, R1, R2, and R3. Suppose that $\mathcal{P}$ and $\mathcal{D}$ are the set of proteins and the set of domains in a given proteome, respectively, where each protein $P_i \in \mathcal{P}$ consists of domains in $\mathcal{D}$.

In a production rule of R1, it is assumed that all domains in protein P_i are created by several mutation events, which rule can be represented as $P_i = \{D_{i_1}, D_{i_2}, \ldots\}$. Then, the cost was defined by

$$cost_{R1}(P_i) = \lceil \log |\mathcal{D}| \rceil \cdot |P_i|, \tag{1}$$

where the base of the logarithm is two, $\lceil x \rceil$ is the integer more than or equal to x, and $|S|$ denotes the number of elements in a set S. It should be noted that P_i can be a multiset, that is, multiple instances of the same domain can be included, and $|P_i|$ is the sum of multiplicities of domains. It takes $\lceil \log |\mathcal{D}| \rceil$ bits to represent the identifier of one domain.

In a production rule of R2, it is assumed that protein P_i is constructed by duplicating P_j and by deletion and insertion of several domains, which rule can be represented as $P_i = P_j - \{D_{j_1}, \ldots\} + \{D_{i_1}, \ldots\}$. Then, the cost was defined by

$$\begin{aligned} &cost_{R2}(P_i; P_j) \\ &= \lceil \log |\mathcal{P}| \rceil + |P_j| \\ &\quad + \lceil \log |\mathcal{D}| \rceil \cdot |P_i - P_j|, \end{aligned} \tag{2}$$

where $S - T$ for multisets S and T denotes the multiset removing elements included in T from S. Among domains of P_j, only the domain that the corresponding bit is one is duplicated.

In a production rule of R3, it is assumed that protein P_i is constructed by duplicating P_j and P_k and by inserting several domains, which rule can be represented as $P_i = P_j + P_k + \{D_{i_1}, \ldots\}$. Then, the cost was defined by

$$\begin{aligned} &cost_{R3}(P_i; P_j, P_k) \\ &= 2 \cdot \lceil \log |\mathcal{P}| \rceil \\ &\quad + \lceil \log |\mathcal{D}| \rceil \cdot |P_i - P_j - P_k|. \end{aligned} \tag{3}$$

Let $r_i (\in \{\text{R1, R2, R3}\})$ be a rule constructing P_i. The size of the grammar $\mathcal{G}$ is represented by

$$|\mathcal{G}| = \sum_{P_i \in \mathcal{P}} cost_{r_i}(P_i). \tag{4}$$

They tried to find a grammar by minimizing the size $|\mathcal{G}|$ through the minimum spanning directed hypertree problem [28]. The problem, however, was intractable for a large number of proteins, and they developed a heuristic method that reduces candidate production rules. Without use of production rules of R3, we can solve the problem in polynomial time for finding the minimum grammar with only R1 and R2 production rules.

2.2 Compression with Domains Ordered

In this study, we deal with only mutation and gene duplication events for compressing sequences of domains because we can find the minimum grammar. Suppose that P_i also represents a sequence of domains. Similar to a production rule of R1 in the previous study, a sequence of domains in protein P_i can be represented by writing identifiers in the same order, that is, its rule can be represented as $P_i = D_{i_1} D_{i_2} \cdots$. Then, the cost that P_i is constructed from domains is equivalent to that in the previous study,

$$cost_{R1}(P_i) = \lceil \log |\mathcal{D}| \rceil \cdot |P_i|. \tag{5}$$

In the case of gene duplication from P_j to P_i, the sequence of domains in P_j are copied, and domains specified by $|P_j|$ bits are deleted. After that, several domains are inserted. There, however, can be several different ways to insert domains into the sequence. For example, consider $P_i = D_2 D_3 D_1$ and $P_j = D_1 D_2 D_1$. To transform P_j into P_i, we may delete $D_1 D_2$ and insert $D_2 D_3$ in front of the sequence. In another way, we may delete the first D_1, and insert D_3 between D_2 and D_1. For our purpose of finding the minimum grammar, we utilize the Levenshtein distance $d_L(P_j, P_i)$ from P_j to P_i, which is defined by the minimum cost of edit operations, insertion, deletion, and substitution [29]. Since domains to be deleted are specified by $|P_j|$ bits, we set the cost of deletion to be zero, and those of insertion and substitution to be one, respectively. Then, the cost that P_i is constructed from P_j is defined by

$$\begin{aligned} &cost_{R2}(P_i; P_j) \\ &= \lceil \log |\mathcal{P}| \rceil + |P_j| + d_L(P_j, P_i) \\ &\quad \cdot (\lceil \log |\mathcal{D}| \rceil + \lceil \log(|P_j| + 1) \rceil), \end{aligned} \tag{6}$$

where $|P_j| + 1$ means the number of positions to be inserted into P_j. In this rule, the position together with the identifier of an inserted domain is specified.

In order to find the minimum grammar consisting of production rules of R1 and R2, we construct a directed graph $G(V, E)$ with a set V of vertices and a set E of directed edges. V consists of v_0 ($P_0 \notin \mathcal{P}$) and v_i for all $P_i \in \mathcal{P}$. E consists of (v_0, v_i) and (v_j, v_i) for all $P_i, P_j \in \mathcal{P}$, where $cost((v_0, v_i)) = cost_{R1}(P_i)$ and $cost((v_j, v_i)) = cost_{R2}(P_i; P_j)$. Then, from the minimum spanning tree of G, we can obtain the minimum grammar $\mathcal{G}$ that constructs the given set of sequences of domains.

3. Results

We used protein domain compositions of seven organisms, *E. coli*, *S. cerevisiae*, *A. thaliana*, *C. elegans*, *D. melanogaster*, *M. musculus*, and *H. sapiens* from UniProt database (Release 2017_07) [30]. For the order of domains in a protein, we used starting and ending positions of each domain in the FT line of UniProt format. The number of proteins and the total number of distinct domains for the seven organisms are shown in Table 1.

We compressed a proteome by considering a protein as a multiset or sequence of domains. Fig. 1 shows the results on the ratio of the compressed size to the original size by considering domains ordered (A) and unordered (B), respectively, and the ratio (A/B) of the compressed sizes for the seven organisms. Table

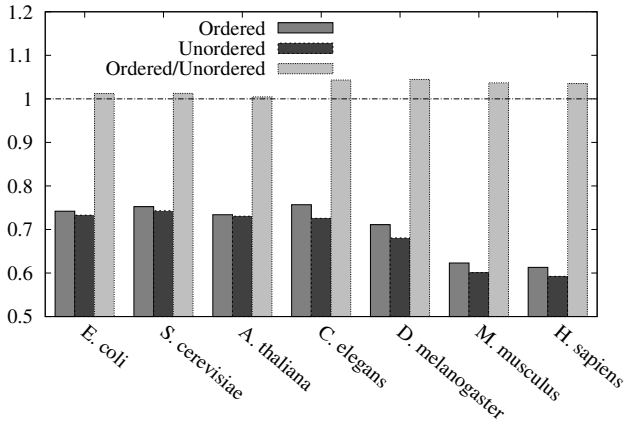

Fig. 1 Results on the ratio of the compressed size to the original size by considering domains ordered (A) and unordered (B), respectively, and the ratio (A/B) of the compressed sizes for organisms, *E. coli*, *S. cerevisiae*, *A. thaliana*, *C. elegans*, *D. melanogaster*, *M. musculus*, and *H. sapiens*.

1 shows the detailed values, where the original size, denoted by 'original', is the sum of costs in the case that all proteins are constructed only by production rules of R1, that is, $\sum_{P_i \in \mathcal{P}} cost_{R1}(P_i)$, and the original size of a proteome with domains unordered is the same as that with domains unordered.

It is seen that the compression ratios of *M. musculus* and *H. sapiens* for proteomes with domains unordered were smaller than those of other organisms as reported in the previous study. Also for proteomes with domains ordered, we can see that the compression ratios of higher organisms were smaller than those of others. It insists that gene duplications have been used more frequently in higher organisms.

For the ratio of the compressed size of a proteome with domains ordered to that with domains unordered, in *E. coli*, *S. cerevisiae*, and *A. thaliana*, the ratio was almost one, and the compressed size for ordered domains was almost the same as that for unordered domains. On the other hand, in *C. elegans*, *D. melanogaster*, *M. musculus*, and *H. sapiens*, the compressed size for ordered domains was about four percent larger than that for unordered domains. It implies that there exist proteins that have almost the same domains and which sequences of domains are largely different. It can be considered that several proteins have changed the functions by replacing domains.

4. Conclusion

In this paper, the grammar for a set of sets of domains was extended, and a grammar to compress a proteome with domains ordered was developed. By finding the minimum grammars of proteomes, we compared evolutionary process of seven organisms, *E. coli*, *S. cerevisiae*, *A. thaliana*, *C. elegans*, *D. melanogaster*, *M. musculus*, and *H. sapiens*. As a result, we confirmed that the compression ratio for higher organisms was smaller than that for others as reported in the previous study. Furthermore, from comparison between the ratios of the compressed size for ordered domains to that for unordered domains, it can be considered in higher organisms that several proteins have changed the functions by replacing domains. As future work, we would like to compress and analyze more organisms, and to obtain comprehensive knowledge for evolutionary processes.

Acknowledgements

This work was partially supported by Grants-in-Aid #16K00392, and #16KT0020 from JSPS, Japan.

References

[1] Grumbach, S. and Tahi, F.: A New Challenge for Compression Algorithms: Genetic Sequences, *Information Processing & Management*, Vol. 30, No. 6, pp. 875–886 (1994).
[2] Rivals, E., Delahaye, J., Dauchet, M. and Delgrange, O.: A Guaranteed Compression Scheme for Repetitive DNA Sequences, *Data Compression Conference* (1996).
[3] Chen, X., Kwong, S. and Li, M.: A compression algorithm for DNA sequences, Vol. 20, pp. 61 – 66 (2001).
[4] Willems, F., Shtarkov, Y. and Tjalkens, T.: The context tree weighting method: basic properties, *IEEE Trans. Inform. Theory*, Vol. IT-41, No. 3, pp. 653–664 (1995).
[5] Lempel, A. and Ziv, J.: On the complexity of finite sequences, *IEEE Trans. Inform. Theory*, Vol. 22, No. 1, pp. 75–81 (1976).
[6] Matsumoto, T., Sadakane, K. and Imai, H.: Biological sequence compression algorithms, *Genome Informatics*, Vol. 11, pp. 43–52 (2000).
[7] Ma, B., Tromp, J. and Li, M.: PatternHunter – faster and more sensitive homology search, *Bioinformatics*, Vol. 18, pp. 440–445 (2002).
[8] Chen, X., Li, M., Ma, B. and Tromp, J.: DNACompress: fast and effective DNA sequence compression, *Bioinformatics*, Vol. 18, No. 12, pp. 1696–1698 (2002).
[9] Behzadi, B. and Fessant, F. L.: DNA compression challenge revisited: A dynamic programming approach, *Combinatorial Pattern Matching*, pp. 190–200 (2005).
[10] Loewenstern, D. and Yianilos, P. N.: Significantly lower entropy estimates for natural DNA sequences, *Computational Biology*, Vol. 6, No. 1, pp. 125–142 (1999).
[11] Korodi, G. and Tabus, I.: An efficient normalized maximum likelihood algorithm for DNA sequence compression, *ACM Trans. Inf. Syst.*, Vol. 23, No. 1, pp. 3–34 (2005).
[12] Nevill-Manning, C. G. and Witten, I. H.: Protein is incompressible, *Data Compression Conference*, pp. 257–266 (1999).
[13] Hategan, A. and Tabus, I.: Protein is compressible, *The 6th Nordic Signal Processing Symposium*, pp. 192–195 (2004).
[14] Cao, M., Dix, T., Allison, L. and Mears, C.: A simple statistical algorithm for biological sequence compression, *Data Compression Conference*, pp. 43–52 (2007).
[15] Pinho, A. and Pratas, D.: MFCompress: A compression tool for FASTA and multi-FASTA data, *Bioinformatics*, Vol. 30, pp. 117–118 (2013).
[16] Mohammed, M., Dutta, A., Bose, T., Chadaram, S. and Mande, S. S.: DELIMINATE–a fast and efficient method for loss-less compression of genomic sequences: sequence analysis, *Bioinformatics*, Vol. 28, No. 19, pp. 2527–2529 (2012).
[17] : available from ⟨http://www.gzip.org/⟩.
[18] : available from ⟨http://www.7-zip.org/sdk.html⟩.
[19] Hach, F., Numanagic, I., Alkan, C. and Sahinalp, S.: SCALCE: Boosting sequence compression algorithms using locally consistent encoding, *Bioinformatics*, Vol. 28, pp. 3051–3057 (2012).
[20] Bonfield, J. and Mahoney, M.: Compression of FASTQ and SAM format sequencing data, *PLoS ONE*, Vol. 8, p. e59190 (2013).
[21] Jones, D., Ruzzo, W., Peng, X. and Katze, M.: Compression of next-generation sequencing reads aided by highly efficient de novo assembly, *Nucleic Acids Res.*, Vol. 40, No. 22, p. e171 (2012).
[22] Deorowicz, S. and Grabowski, S.: Compression of DNA sequence reads in FASTQ format, *Bioinformatics*, Vol. 27, pp. 860–862 (2011).
[23] Hosseini, M., Pratas, D. and Pinho, A. J.: A Survey on Data Compression Methods for Biological Sequences, *Information*, Vol. 7, p. 56 (2016).
[24] Daniels, N. M., Gallant, A., Peng, J., Cowen, L. J., Baym, M. and Berger, B.: Compressive genomics for protein databases, *Bioinformatics*, Vol. 29, No. 13, pp. i283–i290 (2013).
[25] Nag, A. and Karforma, S.: Adaptive dictionary-based compression of protein sequences, *International Journal of Education and Management Engineering*, Vol. 7, No. 5, pp. 1–6 (2017).
[26] Hayashida, M., Ruan, P. and Akutsu, T.: Proteome compression via protein domain compositions, *Methods*, Vol. 67, pp. 380–385 (2014).
[27] Nacher, J. C., Hayashida, M. and Akutsu, T.: The role of internal duplication in the evolution of multi-domain proteins, *BioSystems*,

Table 1 Results on the compressed size by considering domains ordered and unordered for organisms, *E. coli*, *S. cerevisiae*, *A. thaliana*, *C. elegans*, *D. melanogaster*, *M. musculus*, and *H. sapiens*. 'original' denotes the sum of costs in the case that all proteins are constructed only by production rules of R1, that is, $\sum_{P_i \in \mathcal{P}} cost_{R1}(P_i)$, which is equivalent between ordered and unordered cases.

organism	# proteins	# domains	original	ordered (A)		unordered (B)		A/B
E. coli	880	284	18801	13949	0.7419	13777	0.7328	1.012
S. cerevisiae	1296	352	28260	21263	0.7524	21000	0.7431	1.013
A. thaliana	4993	398	111375	81728	0.7338	81349	0.7304	1.005
C. elegans	1104	389	30960	23434	0.7569	22462	0.7255	1.043
D. melanogaster	991	389	29034	20645	0.7111	19762	0.6807	1.045
M. musculus	5820	630	191720	119448	0.6230	115177	0.6008	1.037
H. sapiens	6886	626	226530	138889	0.6131	134170	0.5923	1.035

Vol. 101, No. 2, pp. 127–135 (2010).

[28] Brejová, B., Brown, D. G. and Vinař, T.: Optimal DNA signal recognition models with a fixed amount of intrasignal dependency, *WABI 2003: Algorithms and Bioinformatics: 3rd International Workshop, volume 2812 of Lecture Notes in Bioinformatics*, Springer, pp. 78–94 (2003).

[29] Levenshtein, V.: Binary codes capable of correcting deletions, insertions and reversals, *Doklady Adademii Nauk SSSR*, Vol. 163, No. 4, pp. 845–848 (1965).

[30] The UniProt Consortium: UniProt: the universal protein knowledgebase, *Nucleic Acids Research*, Vol. 45, pp. D158–D169 (2016).