IET Signal Processing

Review Article

IET Journals
The Institution of Engineering and Technology

Application of signal processing for DNA sequence compression

ISSN 1751-9675
Received on 20th August 2018
Revised 18th March 2019
Accepted on 29th April 2019
E-First on 14th June 2019
doi: 10.1049/iet-spr.2018.5392
www.ietdl.org

Bonnie Ngai-Fong Law[1] ✉

[1]*Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong*
✉ *E-mail: ennflaw@polyu.edu.hk*

Abstract: Due to the advancement of high-throughput sequencing technologies, it is now feasible for sequencing individual genomes in a fast and affordable manner. With the significant increase in the number of individual genomes, compression methods are needed to reduce pressure on data storage as well as enable effective data distribution and management. The compression methods can generally be divided into two classes, namely reference-free methods and reference-based methods. In reference-free methods, redundancies within the target DNA sequence to be compressed are explored. In reference-based methods, redundancies between the target DNA sequence and other reference sequences are identified to achieve compression. This type of method is applicable to population sequences which are highly similar to each other and have a small number of mismatches. Some of the methods can also be applied to partially similar sequences such as chromosome sequences or sequences having evolutionary relationship. The authors highlight recent developments in these methods. In the comparative study, the authors' simulation results reveal that the selection of a reference sequence is a crucial factor affecting the compression performance. Use of multiple number of reference sequences and enhancement strategies such as reference rewriting are important to achieve a large compression gain.

1 Introduction

Sequencing techniques allow the DNA data to be represented as a long string made up from four nucleotide bases. This facilitates the computational analysis of our biological make-up for use in areas such as forensics applications, crime investigation or parental connection establishment [1–3]. Recent advancement in next-generation sequencing (NGS) techniques enables sequencing the individual genome in a fast and affordable manner. As a result, a large number of genomic data are produced for various studies. For example, the genomic data has been used for studying variations of genetic diseases on individuals so that personalised medicines and diagnostic tools can be developed [4]. Specific mutations within the genomic data have been used to study the risk of developing certain diseases [5, 6]. The collection of data has also been used to study pattern about an organism's evolutionary history and aid in phylogenetic tree reconstruction [7–10]. On the other hand, NGS techniques have introduced a new challenge in which a lot of DNA sequences have to be stored in various databases [11, 12]. Large public databases for storing DNA sequences include the GenBank at the National Center for Biotechnology Information (NCBI) [13], the European Bioinformatics Institute EMBL database [14] and the DNA Data Bank of Japan (DDBJ) [15]. Records in these three databases are indeed synchronised so that each database contains a copy of the others. Researchers can then download data conveniently from these databases for study.

There are different file formats to store the DNA sequences. Examples include Genbank, EMBL, BAM, SAM, FASTQ and FASTA [16]. These file formats are used in different contexts, but they can be converted to one another easily. Although the exact details to be stored in these file formats may not be the same, they often contain two distinct parts. The first part is metadata such as sequence identifier, annotation and/or description. The second part is the actual nucleotide sequence. Currently, only general-purpose lossless compression methods such as gzip and bzip2 [17] are applied to reduce the storage.

From 1982 to now, the number of the nucleotide bases in GenBank is doubled approximately every 1.5 years [18]. In projects such as the 1000-Genomes project [19] or the UK 10k Project [20], thousands of genomes are sequenced to facilitate the study of the genomic landscape of various diseases [5]. Consider that the size of the genome sequence data of one person reaches several gigabytes, a large storage is required to store the complete genome sequences generated from these projects. To address these issues, effective compression algorithms are needed for DNA sequences.

Due to the different characteristics between the metadata and the DNA nucleotide sequence, specific compressor for a particular file format always adopts different principles in compressing the metadata and the nucleotide sequence. For example, the compressor for FASTQ adopts template compression method for the header, LZ coding for the quality line and a combination of substitution-based method and Huffman coding for nucleotide sequence [21]. In this paper, we do not focus on any specific format for storing DNA data. Rather, our focus is on the compression of nucleotide sequences. Hence, the reviewed methods can be applied to all types of file formats in which the nucleotide sequence is part of the format. Readers may refer to [16, 17] for surveys of approaches for compressing different file formats.

As mentioned in [22], the genome sequence for an individual containing Adenine (A), Thymine (T), Cytosine (C) and Guanine (G) is almost incompressible. Thus, special DNA sequence characteristics have to be explored in its compression. The basic idea in traditional DNA nucleotide sequence compression is to find identical sub-sequences within the target DNA sequence to be compressed so that they are encoded only once. Examples of popular compression methods in this group of techniques include BioCompress [23, 24], Cfact [25], GenCompress [26–28], CTW + LZ [29] DNACompress [30], DNASequitur [31] and GeNML [32, 33]. While these tools use different ways to identify and characterise intra-sequence similarities, the compression rate is not high. Recently, statistical methods have been proposed to explore redundancies within the DNA sequence to be compressed. Various models are used to predict the occurrence probability so as to achieve compression. Examples include XM [34], GENBIT [35], HuffBit [36], DNABIT [37], PRDNAC [38], SeqCompress [39], DNABIT-2 [40] and the modified Huffman coding [41]. For these reference-free methods that explore redundancies within the DNA sequences, the storage size in most cases can be reduced by <40%.

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Base	C	T	T	G	G	A	G	C	C	C	C	G	C	G	C	C	C	A	C	A	G	C	T	T	T	G	G	C	C	T
Position	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
Base	C	C	G	G	T	T	C	C	A	T	C	G	C	T	G	C	C	C	T	T	G	T	A	G	G	G	A	T	C	C

a

01	11	11	10	10	00	10	01	01	01	01	10	01	10	01	01	01	00	01	00	10	01	11	11	11	10	10	01	01	11
01	01	10	10	11	11	01	01	00	11	01	10	01	11	10	01	01	01	11	11	10	11	00	10	10	10	00	11	01	01	

b

C	T	T	G	G	A	G	C	C	C	C	G	C	A	C	A	G	C	T	T	T	G	G	C	C	T	C	C	G	G	
T	T	C	C	A	T	C	G	C	T	G	C	C	C	T	T	G	T	A	G	G	G	A	T	C	C					

c

Fig. 1 *A simple example illustrating DNA sequence representation*

(a) Short segment of the chromosome 19 of the human Homo sapiens genome, ***(b)*** Its lossless encoded representation with 'A' represented as '00', 'T' as '11', 'C' as '01' and 'G' as '10', ***(c)*** Resultant segment after the first referencing instruction is applied

It is certainly insufficient for large-scale data distribution and storage.

For the past decade, different effective reference-based compression algorithms have been proposed. Instead of considering only redundancies within the target sequence to be compressed, similarities among different DNA sequences are considered. This is often termed 'inter-sequence similarities'. For example, similarities among different chromosome sequences of the same species [42, 43], similarities between sequences that may be related through evolution [44] and similarities between individual genomes within a population of the same species [17, 22, 45] can be considered. Compression can then be achieved through referencing instructions. In other words, similar sub-sequences are encoded once only which act as references to their occurrence at other locations of the same sequence or other sequences. Examples of methods in this group include DNAzip [12], RLCSA [46], RLZ [47, 48], GRS [49], GReEn [50], iDoComp [51], COMRAD [52, 53], ERGC [54], CoGI [55], MSC [56], GDC [57, 58], FRESCO [59] and RCC [60]. With the use of appropriate reference sequences, the storage size reduction in some cases can be over 90%.

The objective of this paper is to present an up-to-date account of recent research done in DNA sequence compression. Existing DNA sequence compression surveys focus mainly on those algorithms using intra-sequence similarities [61–63]. Our focus is on the recent development of compression algorithms considering not only intra-sequence similarities, but also statistical redundancies and inter-sequence similarities. We surveyed publications on DNA compression mostly from 2010 to 2018. This paper addresses the basic rationales, techniques and their important results. This study highlights the principles of different compression algorithms. Besides, we have performed a comparative study of compression methods in terms of the compression gain and the computational requirements for two population datasets. It will be beneficial to individuals who are interested in DNA compression as well as general signal processing researchers who may want to know more about DNA compression methods.

This paper is organised as follows. We will first give an introduction to DNA sequences and describe their characteristics in Section 2. After that, important properties of DNA sequences and how these properties can be characterised in a computational manner will be discussed in Section 3. Then descriptions about two kinds of compression methods are given. In particular, Section 4 describes methods that explore redundancies within a DNA sequence while Section 5 considers those reference-based methods for highly similar population sequences. Some of the reference-based methods can be extended to work on partially similar datasets and are discussed in Section 6. Finally, Section 7 concludes the paper.

2 Characteristics of DNA sequences

Biological information of living organism is encoded in the long DNA sequence which is made up from only four different nucleotide bases, namely A, T, C and G. Fig. 1 shows a short 60 bases sub-sequence from the chromosome 19 of the human Homo sapiens genome. From a signal processing point of view, we can simply consider DNA to be a long string consisting of four different 'symbols'. Hence, without compression, 2 bits are needed to encode one symbol in a lossless manner. By encoding the base 'A' as '00', 'T' as '11', 'C' as '01' and 'G' as '10', the encoded sequence requires 120 bits as shown in Fig. 1*b*.

Typically, a DNA sequence contains bases in the order of millions. For example, the human genome has around 3 billion bases while yeast has around 12 million bases. Hence, the storage will be around 0.75 billion bytes for a human genome and 3 million bytes for a yeast sequence without any compression. Although DNA should not be a random sequence, the occurrence frequencies of the four bases are rather uniform. General purpose compression algorithm cannot compress the sequence. In some cases, these general purpose compression algorithms even expand the sequence in the sense that more than 2 bits are used to encode a base. Certainly, DNA sequence characteristics have to be considered to achieve compression.

2.1 Intra-sequence repetitions

Studies find that there are many repetitions in DNA sequences. It is estimated that the human genome has around 50% of repeat sequences within itself [64]. For example, the human genome contains over 100 million copies of the Alu repeat which is about 300 bases long [65]. All modern mammals contain a highly repeated family of long interspersed repeated DNA called the L1 family [66] which is made up from >6000 bases. Hence, these sub-sequences sharing similar bases compositions and arrangement can be explored for compression purpose.

To illustrate the idea, consider the short segment of the chromosome 19 of the human Homo sapiens sequence in Fig. 1*a*. The sub-sequence 'CTTG' (highlighted by the green colour) can be found at two different base positions, the first occurrence from positions 1 to 4 and the second occurrence from positions 48 to 51. Similarly, 'GCCC' can be found at three different locations (highlighted by the orange colour): first from positions 7 to 10, second from positions 14 to 17 and third from positions 45 to 48. Hence, this kind of intra-sequence similarities can be explored to achieve compression. In particular, repeats are encoded only once.

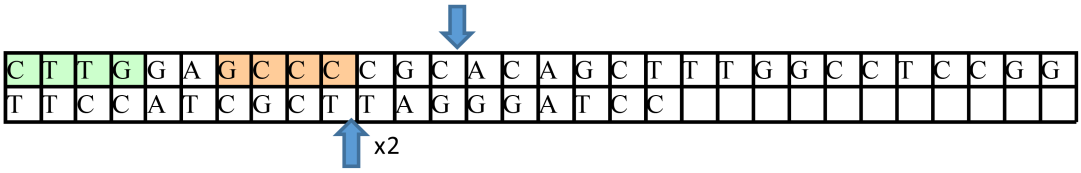

Fig. 2 *With consideration of three referencing instructions at positions shown using blue arrows, the number of bases to be encoded reduces to 49*

Its occurrence at other locations can be constructed through using referencing instructions. Consider 'GCCC' which is found from positions 7 to 10 (highlighted by the orange colour in Fig. 1*a*), a referencing instruction can be used to denote its second occurrence from positions 14 to 17 as follows:

Referencing instruction at position 14:
sub-sequence with (start: 7, end: 10).

In this way, the second occurrence of 'GCCC' can be removed from the original sequence in Fig. 1*a*. Fig. 1*c* shows the resultant sub-sequence that needs to be encoded. Through using the referencing instruction, the number of bases that needs to be encoded then reduces from 60 to 56. Similarly, the following two referencing instructions can be added to remove the remaining two similar sub-sequences 'GCCC' and 'TTG' in the last row of Fig. 1*a*.

Referencing instruction at position 45:
sub-sequence with (start: 7, end: 10).
Referencing instruction at position 49:
sub-sequence with (start: 2, end: 4).

After having these two referencing instructions, we only need to encode 49 bases as shown in Fig. 2 instead of 60 bases. The referencing instructions, however, need to be encoded to achieve lossless compression. A list containing the starting and the ending positions of all the referencing instructions can be constructed. The list can then be encoded further using methods such as arithmetic coding.

The above illustration considers only exact repetitions. In fact, more repeats can be found if approximate repeats are also explored. The approximate repeats mean that the two matched sub-sequences have similar base compositions except at a few base positions. These mismatched bases can be characterised through substitution, deletion and/or insertion operations. For example, 'GCCT' (highlighted in blue colour in Fig. 1*a*) is similar to 'GCCC' except at the last base position. In this case, we can perform the substitution operation at the last position to replace 'C' by 'T'. In approximate repeats, in addition to the starting and the ending positions of repeats, the referencing instructions should also contain:

- the types of operations to be performed such as substitution, insertion and deletion;
- the relative positions at where the bases are different in the approximate repeats; and
- the replacement base for a substitution operation and the inserted base for an insertion operation.

The information can be concatenated and compressed using methods such as arithmetic coding. As long as similar sub-sequences are long, it would be advantageous in terms of compression to perform approximate matching.

DNA sequence contains a special kind of repetition called complementary repeat or palindrome. Due to its double helix structure, base in one DNA strand would bind to its complementary base in the other DNA strand. The base 'A' is complementary to 'T' while 'C' is complementary to 'G'. Hence, the sub-sequence 'GCCC' is complementary repeat of 'GGGC' as the complement bases of 'GCCC' are 'CGGG' and their reverse ordering are 'GGGC'. Similarly, 'GCCT' is complementary repeat of 'AGGC'. By considering the approximate complementary repeats, we can see that the sub-sequence 'GGGA' (purple colour in Fig. 1*a*) is complementary repeat of 'GCCC' with the last base substituted by 'A'. Thus sub-sequence 'GGGA' can be represented through referencing and substitution operations. In summary, all types of repeats including the exact repeat, the approximate repeat and the complementary repeat can be explored in DNA sequences to achieve compression.

2.2 Inter-sequence repetitions

In addition to intra-sequence similarities, DNA sequences contain other kinds of similarities. For example, different chromosome sequences of a species can be similar to each other. Genomes of different individuals in the same species can have high similarities as well. To illustrate these kinds of inter-sequence similarities, the short segment of the chromosome 19 of the human Homo sapiens in Fig. 1*a* is considered. Figs. 3 and 4 show, respectively, the sub-sequences in the chromosome 19 that can be found in different chromosome sequences of the human Homo sapiens and the two chromosome sequences of the mouse genome. In Fig. 3*a*, a sub-sequence from positions 38 to 57 of the chromosome 19 is essentially same as that of the chromosome 3 except at position 51 where the base 'G' in the chromosome 19 should be substituted by 'A' to obtain the corresponding sub-sequence in the chromosome 3. Similarly, from Figs. 3*b–d*, we can see that chromosomes 5, 8 and 11 contain approximate repeats of sub-sequences in the chromosome 19 with a few number of mismatched bases. Figs. 4*a* and *b* show the similarities between the chromosome 19 of the human Homo sapiens and chromosomes 5 and 16 of the mouse genome. With the use of referencing instructions similar to the case of intra-sequence similarities, part of the sequence can be encoded by referencing to the chromosome 19 of the human Homo sapiens for all these chromosome sequences. This example shows that similarities can commonly be found from other sequences in addition to it.

3 Signal processing techniques for identifying sequence similarities

Existing DNA-oriented compression methods are based on the ideas of finding various types of repeats in DNA sequences. In fact, searching of these repeats can be time-consuming. It is often not a trivial task to find all these repeats in a very long DNA sequence efficiently and effectively. Various methods have been proposed for identifying similar sub-sequences. These methods can generally be grouped into five classes, namely suffix-based array methods, optimisation methods through dynamic programming, seed extension methods, rule-based methods and parsing methods. A brief description of these methods is given here for completeness. For detailed discussion, readers may refer to previous survey papers on DNA compression methods [61–63].

A suffix tree is a kind of data structure that is used to characterise exact matches. In suffix-based methods, repeats are considered as common prefixes and a suffix tree is constructed so that similar sub-sequences are represented as the longest common prefix. Compression methods such as Cfact [25], RLCSA [46], iDoComp [51] and FRESCO [59] have used suffix tree-based methods for identifying and characterising repeats.

The identification of similar sub-sequences can also be formulated as an optimisation problem which is solved iteratively through dynamic programming. The compressed size can be considered as a cost function for minimisation. It has been used in methods such as GenCompress [26–28] and CTW + LZ [29]. However, the computational complexity of dynamic programming is often very high. To reduce the complexity, various seed-based

Position	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57
Chr 19	C	A	T	C	G	C	T	G	C	C	C	T	T	G	T	A	G	G	G	A
Chr 3	C	A	T	C	G	C	T	G	C	C	C	T	T	A	T	A	G	G	G	A

a

Position	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
Chr 19	A	C	A	G	C	T	T	T	G	G	C	C	T	C	C	G	G	T	T	C	C	A	T	C	G	C	T	G
Chr 5	A	C	A	G	C	T	T	T	G	G	C	C	T	C	A	G	T	T	T	C	C	T	T	A	G	C	T	G

b

Position	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
Chr 19	G	C	C	C	A	C	A	G	C	T	T	T	G	G	C	C	T	C	C	G
Chr 8	G	C	C	C	A	C	A	G	C	T	T	T	G	T	C	C	T	C	C	G

c

Position	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
Chr 19	C	C	C	G	C	G	C	C	C	A	C	A	G	C	T	T	T	G	G	C	C	T	C	C	G	G
Chr 11	C	C	C	G	C	G	C	C	C	A	C	A	G	C	C	T	G	G	G	C	A	T	C	C	G	G

d

Fig. 3 *Similar sub-sequences among the chromosome 19*

(a) Chromosome 3, ***(b)*** Chromosome 5, ***(c)*** Chromosome 8, ***(d)*** Chromosome 11 of the human Homo sapiens. The highlighted entries show the different base compositions at certain positions

Position	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
Chr 19	G	C	C	C	C	G	C	G	C	C	C	A	C	A	G	C	T	T	T	G	G	C	C	T	C	C	G	G
A	G	C	C	C	C	G	C	G	C	C	C	T	C	A	G	C	T	T	T	G	T	C	C	T	T	C	G	G

a

Position	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Chr 19	T	G	G	A	G	C	C	C	C	G	C	G	C	C	C	A	C	A	G	C	T	T	T	G	G	C	C	T	C	C
B	T	G	G	A	-	C	C	C	A	G	C	T	C	C	C	A	C	A	G	C	T	T	T	G	C	C	C	T	C	C
Position	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53		54	55	56	57	58	59	60	
Chr 19	G	G	T	T	C	C	A	T	C	G	C	T	G	C	C	C	T	T	G	T	A		G	G	G	A	T	C	C	
B	G	-	-	-	-	-	A	T	C	G	C	T	G	C	C	C	T	T	G	T	A	A	G	G	G	A	T	C	C	

b

Fig. 4 *Similar sub-sequences among the chromosome 19 of the human Homo sapiens*

(a) Mus musculus strain C57BL/6J chromosome 5, GRCm38.p4 C57BL/6J (denoted as A), ***(b)*** Mus musculus strain C57BL/6J chromosome 16, GRCm38.p4 C57BL/6J (denoted as B). The highlighted entries show the different base compositions at certain positions

methods can be used. Short seed matches considering only sub-sequence pairs with a small fixed length are first searched. These short matches are then extended to longer sub-sequence matches progressively. Seed-based methods are used frequently in DNA compression methods. Examples include DNACompress [30], MSC [56] and RCC [60].

The rule-based methods use grammar rules to characterise repeats. Consider a short DNA sequence $S = \{ACTGCTACTG\}$. S can be represented as $R_1R_2R_1$ where the two rules are defined as $R_1 \rightarrow AR_2G$ and $R_2 \rightarrow CT$. These rules are then entropy encoded. Example rule-based compression methods are DNA Sequitur [31] and COMRAD [52, 53]. In parsing methods, the DNA sequence is usually divided into a number of phrases using a greedy strategy where the phrases denote repeats in the sequence. Parsing methods have been used in RLZ and RLZ-opt [47, 48] and LZ77 [67].

In fact, these different methods can be combined to search for repeats. For example, GDC [57, 58] has used both seed-based methods and parsing methods for repeats identification. Besides, several homology search engines are available for searching repeats in DNA sequences. Examples include Blastn [68] and PatternHunter [69]. Blastn stands for the basic local alignment search tool for nucleotides. It is freely available online [70].

4 Reference-free DNA compression methods

In reference-free compression methods, the DNA sequence is compressed through various ways of exploring the redundant information within the sequence itself. Table 1 summarises these reference-free compression algorithms by listing their publication year, the types of redundancies explored, the characteristics of the compression method as well as the bits per base (bpb) performance. Traditionally, the redundant information is the repetitive sub-sequences existed within the DNA sequence itself. This type of redundancy has been explored by many DNA compression algorithms. They generally use the five different classes of methods discussed in Section 3 (i.e. suffix-based methods, dynamic programming, seed-based methods, rule-based methods and parsing methods) to identify and represent the similar sub-sequences found within the target DNA sequence to be compressed. The repeats are encoded by referencing to its previous occurrence while the non-repetitive part is encoded using arithmetic coding, context-tree weighting (CTW) or the 2-bit lossless representation.

Starting from 2007, statistical methods are getting popular for reference-free DNA sequence compression. Some of these methods develop models to identify the unbalanced symbol occurrence

Table 1 Summary of reference-free DNA sequence compression methods that explore redundancies within the DNA sequence

Publication year	Name	Redundancies considered	Characteristics of the compression method	bpb
1993, 1994	BioCompress [23, 24]	Exact match	One of the early approaches for DNA compression. A sub-sequence is encoded by referencing to an identical sub-sequence occurring in the past. Only the position of the previously occurred similar sub-sequence and the repetition length are encoded. BioCompress-1 uses 2 bpb for non-repetitive sequences while BioCompress-2 uses arithmetic coding.	1.785
1996	Cfact [25]	Exact match	Uses suffix tree to characterise the repeats. The suffix tree is built in the first pass while LZ-based coding is done in the second pass. As building the suffix tree is time-consuming, the compression time of Cfact is longer than that of BioCompress.	Around 1.8
1999, 2000	GenCompress [26–28]	Exact and approximate match	Uses dynamic programming method to identify repeats. In approximate repeats, substitutions are considered in GenCompress-1 while deletions and insertions are also explored in GenCompress-2.	1.7428
2000	CTW + LZ [29]	Exact and approximate match	Uses dynamic programming method to identify repeats. Long repeats are encoded by a LZ77-based algorithm while short repeats are compressed using CTW (context-tree weighting).	1.7389
2002	DNACompress [30]	Exact and approximate match	Uses short seed matches method to find repeats. In particular, a tool called PatternHunter is used for searching repeats. Then repeats and non-repeat regions are encoded using LZ-based compression scheme.	1.7254
2004	DNASequitur [31]	Exact and approximate match	Uses grammar-based method to characterise repeats.	2.12
2005	GeNML [32, 33]	Exact match and substitutions in approximate match	Fixed-size blocks are encoded by referencing a previously encoded sub-sequence with a minimum Hamming distance.	1.6882
2007	XM [34]	Statistical method	A statistical approach that attempts to use a shortcode to represent a frequent symbol. A mixture of models is used for predicting the occurrence probability of a base. Adaptive coding is then used for encoding the predictions.	1.6560
2010	GENBIT [35]	Statistical method	A segment containing four bases is replaced by an 8-bit binary number. If the consecutive segments are the same, a specific bit '1' is introduced as the ninth bit to achieve compression.	2.2335
2010	HuffBit [36]	Statistical method	Extended binary trees are constructed for compression.	NA
2011	DNABIT [37]	Exact match (block-based)	Binary bits are assigned to exact repetitive fragments of DNA sequences. The sequence is sub-divided into blocks and compressed by referencing to its previous occurrence.	Around 1.6
2012	PRDNAC [38]	Exact match	A symbol table about the repeats and a table about the reverse complement repeats are formed and encoded. The procedure of forming the tables is performed iteratively until an optimum compression is achieved.	1.5170
2014	SeqCompress [39]	Statistical method	Statistical models and arithmetic coding are used to achieve compression. The statistical model is developed to find frequent repeats of certain length. They are then encoded using arithmetic coding.	NA
2014	Runlength + ASCII [71]	Statistical method	A modified run-length coding is applied which is then represented using ASCII values.	1.6860
2014	DNAC-K [72]	Clustering of sub-sequences	The DNA sequence is divided into a number of sub-sequences. These sub-sequences are clustered iteratively to group similar sub-sequences together. The result of clustering is encoded by Huffman coding.	1.6620
2016	Optimal seed [73]	Exact and approximate match	The seed based method is used to identify repeats. These repeats are then stored onto a dictionary which is encoded together with the parsed sequence.	1.6190
2016	Runlength + Huffman + ASCII [74]	Statistical method	Each base is represented using 2 bits. A modified run length encoding is then applied. This is followed by Huffman coding. Finally, the encoded sequence is converted into ASCII values.	1.3740
2016	DNABIT-2 [40]	Exact blocks match	Multiple bit pattern representations are developed to encode repetitive block patterns. These representations are then encoded using ASCII characters.	1.5930
2017	Context modelling approach [75]	Exact and approximate match	Context models among adjacent bases with different orders are obtained. These models are then combined through weighting to obtain one coding model for compression.	Around 1.7
2018	Modified DNABIT [76]	Exact blocks match	The original DNABIT method is modified to use extended ASCII encoding to further compress the encoded sequence.	1.4330
2018	Modified HuffBit [77]	Statistical method	Huffman coding is applied to encode the extended binary tree. Each base is then replaced by binary bits generated from the tree.	NA
2018	Modified Huffman coding [41]	Statistical method	By analysing frequent repeats, multiple 'skewed' Huffman trees are developed to achieve compression.	NA

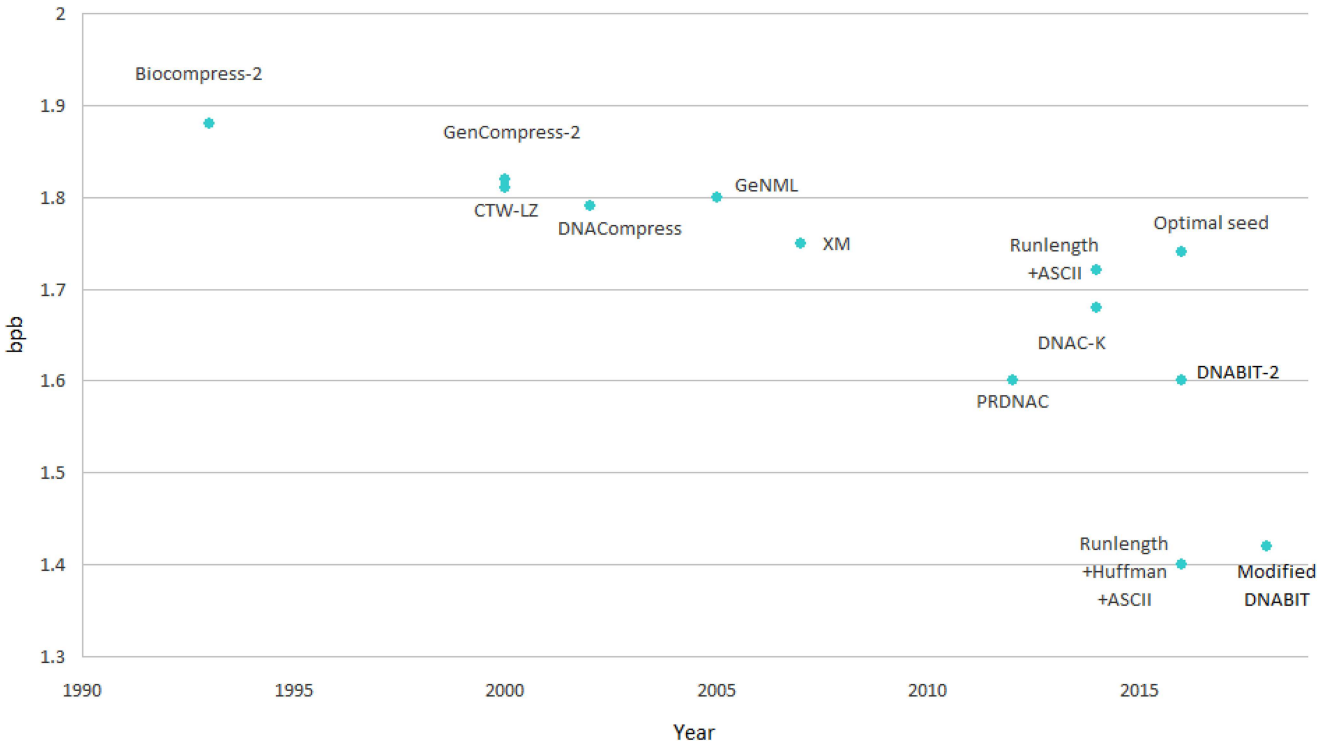

Fig. 5 *bpb performance of various reference-free compression methods against the publication year*

frequencies and predict their occurrence probabilities. In some cases, the 2-bit lossless representation is used to represent a base. The resultant binary sequence is then compressed through run-length coding, Huffman coding and arithmetic coding. The encoded sequence can further be compressed by using ASCII representation. To reduce the computational complexity, a long DNA sequence can also be divided into blocks (i.e. short segments) before statistical methods are applied. These blocks can be clustered to group similar ones together for compression.

Source codes of these algorithms are generally unavailable. Most researchers use standard benchmark DNA sequences from http://www.cs.tut.fi/~tabus/genml/ to compare their work. Through collecting the published results available in the literature, Fig. 5 is drawn to study the bpb performance of these different methods. Without compression, 2 bits are required for encoding a base. Through identifying redundancies within the DNA sequence, all these methods could reduce the bpb to be smaller than 2. It is interesting to see that statistical-based methods proposed recently (such as the modified DNABIT [76]) generally have better performance than methods exploring intra-sequence similarities (such as the optimal seed method [73]).

From 1993 to 2018, the bpb drops from around 1.88 to 1.42. This implies that the compression ratio increases from 1.06 to 1.41 where the compression ratio is defined as the ratio between the uncompressed bpb and the compressed bpb values. In contrast to the development in multimedia data compression, images can always be compressed two or three times losslessly. The compression performance of DNA sequence needs further improvement before large scale distribution and management of this kind of data is possible.

5 Reference-based DNA compression methods for population sequences

Due to the advancement in NGS, population sequences are abundant [11, 19, 20, 78, 79]. Population sequences contain DNA sequences collected from different individuals within a population of the same species. The most recent development in DNA compression indeed targets at population sequences. Studies find that sequences within the same species are highly similar among themselves. For example, only ~0.1% of the 3 GB human genome is specific to an individual. The other 99.9% is shared among all people [57]. Therefore, reference-based compression methods can be used to explore this kind of redundancy for effective compression.

Consider an example of a set of population sequences consisting of 3615 Home sapiens mitochondrial sequences. By comparing these 3615 sequences with the revised Cambridge reference sequence (GenBank accession number of AC_000031) [80], the average number of different bases between a DNA sequence within the population and the revised Cambridge reference sequence is only 33.8. Out of the sequence length of around 16,000 bases, the average similarity in bases is over 99.7%. Hence, it will be highly effective if these sequences are compressed by referencing to the revised Cambridge sequence so that only the base differences are encoded. This forms the basic assumption in all the recent population-based DNA compression algorithms exploring inter-sequence similarities.

In this part, compression methods for population sequences are reviewed in Section 5.1. Then our comparative study of these compression methods in terms of both the bpb performance and computational concerns is presented in Section 5.2.

5.1 Compression methods for population sequences

Table 2 summarises the DNA compression methods for population sequences. We can see that all these methods were proposed within the last ten years. Due to the highly similar nature of sequences within the population dataset, some methods simply encode the base-to-base difference with respect to a reference sequence. In some other approaches, inter-sequence similarities are searched using the five classes of techniques outlined in Section 3.

As shown in Table 2, all the compression methods for population sequences rely on the use of a highly similar reference to achieve compression. The reference sequence needs to be a good representation of all sequences within the population and should be highly similar to all of these sequences. As shown in [22, 59], a change of the reference sequence can have a big impact on the compression performance.

We also believe that one key factor affecting the performance is the way to select the reference sequence. Based on the differences in reference selection, we classify these algorithms into three groups. The first group uses one fixed genome as the reference sequence. The second group enhances the choice of the reference sequence through either constructing a consensus sequence or using more than one reference sequences. The third group uses

various heuristics to define one or multiple number of reference sequences which is then modified and enhanced progressively to further improve compression.

Through collecting the published results available in the literature, the bpb performance of some of the reference-based compression methods for the dataset *S. paradoxus* [60] is studied as shown in Fig. 6. Comparing the reference-based methods in Fig. 6 with the reference-free methods in Fig. 5, we can see that the bpb for reference-based methods is in the range of 0.11–0.80. In contrast, the bpb for reference-free methods is in the range of 1.40–1.88. This shows that the use of a reference sequence can potentially improve the bpb. For compression methods in Group 1, the reference is specified by the user with prior knowledge on the dataset. Some methods choose an external genome as the reference while others choose one sequence within the population as the reference. Ideally, a way to find the best reference sequence is to perform compression by selecting one sequence within the population as the reference sequence one by one. The one that yields the lowest bpb can then be selected as the best reference sequence. However, it would be very time-consuming and might not be practical for large datasets. Hence, the performance of Group 1 methods depends greatly on the ability of the user in finding a suitable reference sequence.

Selecting one sequence as the reference might not be optimal as this sequence might not be highly similar to all sequences within the population simultaneously. For compression methods in Group 2, the reference sequence can be constructed by considering base compositions of all the sequences within the population or by using repeats found within sequences in the population. The number of reference sequences can also be more than one, i.e. a set of

Table 2 Summary of DNA sequence compression methods for population sequences. These algorithms are divided into three groups, depending on how the reference sequence is defined

Year	Name	Compression method	Choice of reference sequence
		Group 1	
2009	DNAzip [12] http://www.ics.uci.edu/~dnazip/	Base to base difference between a DNA sequence and a reference sequence is identified. The difference is then encoded in the form of single nucleotide polymorphisms (i.e. substitution operation), or insertions and deletions of multiple consecutive bases.	One external genome is used as the reference.
2010	Run length compressed suffix array (RLCSA) [46]	Compressed suffix array is adapted to compress a group of highly similar sequences together. Run length coding is then applied to the compressed suffix array.	The reference sequence is selected to be one of the DNA sequences inside the population dataset.
2010, 2011	Relative Lempel–Ziv (RLZ) [47, 48]	Lempel–ziv (LZ77) approach for general data compression is applied for parsing the DNA sequence into substrings. The compression is then done by encoding the sub-sequence with respect to a reference sequence. Correlations between the starting points of the sub-sequences and their positions in the reference sequence are exploited to further improve the compression performance.	The reference sequence is selected to be one of the DNA sequences inside the population dataset.
2011	Genome ReSequencing (GRS) [49]	Difference between a target sequence to be compressed and a reference sequence is evaluated. The longest common sub-sequence is obtained and the part of the sub-sequence that is different from the reference is extracted and compressed by Huffman coding.	One reference sequence is selected by the user.
2012	Genome Resequencing Encoding (GReEn) [50] http://bioinformatics.ua.pt/software/green/	Two models are considered in compression: a copy model for similar sub-sequences and a static model for non-repeats. The copy model achieves compression through using a pointer to a position in the reference sequence that has high probability of containing a repeat.	One genome sequence is chosen as the reference sequence.
2015	iDoComp [51] https://github.com/mikelhernaez/iDoComp	The representation in FRESCO is adapted to consider substitution and insertion operations in each approximate sub-sequence match. An adaptive arithmetic coding is then used to compress the approximated matches.	The reference sequence is selected to be one of the DNA sequences inside the population dataset.
		Group 2	
2009	Reference-based compression approach [45]	Variations from a reference sequence including single nucleotide polymorphisms and contiguous insertions and deletions are encoded.	The reference sequence is a consensus sequence constructed by considering statistics at each base position of all the DNA sequences inside the population dataset.
2012, 2015	Compression using Redundancy of DNA (COMRAD) [52, 53] https://sourceforge.net/projects/comradmpi/files/COMRADMPI/	A dictionary that stores repeats of all sequences in the population dataset is constructed iteratively. The repeats are encoded into short words to achieve compression.	The reference sequence is constructed based on the dictionary of repeats.
2015	Referential Compression Algorithm [81]	The longest matching prefix of the sequence is replaced by the matches present in the reference set to achieve compression.	The reference set is constructed from similar sub-sequences found in a set of randomly selected sequences in the population dataset.
2015	ERGC (Efficient Referential Genome Compressor) [54]	Sequences are divided into segments. Hashing is used to match repeats in these segments. Finally, repeats and non-repeat regions are fed into the PPMD (prediction by partial matching) compressor for further compression.	The reference set is constructed from fixed-sized repeats found within sequences in the population dataset.

Year	Name	Compression method	Choice of reference sequence
2015	CoGI [55] http://admis.fudan.edu.cn/projects/cogi.htm	Sequences are converted into bit-streams using a static coding scheme. Exclusive-or operations is then applied between a sequence and the reference sequence. The whole bit-streams are then re-arranged in a matrix form like binary images which are compressed by a rectangular partition coding method.	The reference sequence is selected using techniques based on co-occurrence entropy and multi-scale entropy.
2015	MSC [56] http://www.eie.polyu.edu.hk/~nflaw/DNAComp/index.html	Approximately matched sub-sequences of all sequences in the population are performed. These matched sub-sequences are then encoded together to achieve compression.	Through approximate sub-sequence matches of all sequences inside the population, the concept of reference sequence is based on selecting appropriate sets of sequences that have been compressed during the progressive compression.
2017	Reference-based Inter Chromosomal Similarity compression [82]	DNA sequences are partitioned into pre-defined sized blocks. Blocks are then coded using dictionary-based coding method.	The reference set is constructed from fixed-sized repeats found within the chromosome sequences.
		Group 3	
2011, 2015	Genome Different Compressor (GDC) [47, 58]	LZ-77 is used for compression where short seed matches with a number of single base mismatches are considered before long seed matches. Parsing of the target sequence into the reference sequence is performed through hashing.	The reference sequence is selected as the one which has the maximum number of distinctive sub-sequences within the population. The reference is then extended by adding to it any significant long sub-sequence that cannot be matched during progressive compression of sequences inside the population dataset.
2013	A Framework for Referential Sequence Compression (FRESCO) [59] https://github.com/hubsw/FRESCO	A reference based compression method in which a mutated base is combined with its preceding exact sub-sequence matches to achieve compression. Besides, suffix tree of the reference sequence is constructed to match prefixes of a target sequence to be compressed with sub-sequences of the reference.	The reference sequence is constructed based on single mutations. Two methods are used to select and/or enhance the reference sequence: 1. It is selected by analysing a reference-based compression performance with an arbitrarily chosen reference sequence taken from the population dataset. 2. The reference sequence is rewritten by analysing the most frequently occurring mismatches in the sub-sequence matches.
2018	RCC [60] http://www.eie.polyu.edu.hk/~nflaw/RCC/index.html	Feature-based clustering is first performed to group similar sequences inside the population into a number of clusters. Then one reference is constructed for each cluster to perform reference-based compression within that cluster. Second-level reference-based compression is performed for all reference sequences in different clusters.	A number of reference sequences are constructed which depend on the number of clusters found in the population. The reference sequence is constructed by analysing the statistics of variants obtained from sub-sequence matches.

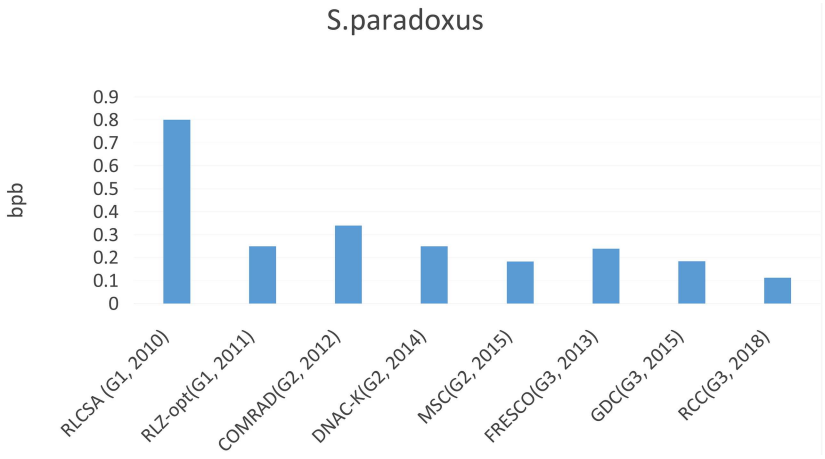

Fig. 6 *bpb performance of various compression methods for the S. paradoxus dataset*

sequences that has been compressed during the progressive compression can be used as references. The flexibility of selecting and constructing the reference sequence can improve the overall compression in the population dataset. Generally, the bpb values in methods in Group 2 are lower than those in Group 1 as depicted in Fig. 6.

To further enhance the performance, methods in the third group enhance the selected/constructed reference sequence by using knowledge such as the most frequently occurring mismatches or any significantly long sub-sequence that cannot be matched. In this way, the reference sequence could be modified progressively when sequences in the population dataset are compressed. As shown in Fig. 6, the bpb performance of Group 3 methods is generally better than that of Group 2 methods.

Irrespective of how the reference sequence is selected, most of these methods rely on the use of a single reference sequence in the

compression. Essentially, it is assumed that a single reference sequence is sufficiently representative for all sequences inside the population. This, in fact, might not be true. For example, there are big variations between the southern group and the northern group of Han Chinese. One reference sequence may not provide good representation for both southern and northern groups at the same time. Hence in RCC, feature-based clustering is first performed to sequences within the population before applying reference-based compression. In this way, different reference sequences can be used in different clusters to describe the sub-structure found in particular groups within a population. With the use of multiple reference sequences, RCC gives the smallest bpb as shown in Fig. 6.

5.2 Comparative studies

To demonstrate the effect of the choice of reference sequence on the compression performance, we have performed a comparative analysis of some recent methods in Groups 1–3. In Group 1, where the reference sequence is pre-selected, RLCSA, RLZ-opt and iDoComp are chosen. In Group 2, MSC is considered in which the reference is based on the idea of sub-sequence matches among different DNA sequences within the population. In Group 3, GDC, FRESCO and RCC are tested. These three methods involve modifying the reference through adding extra phrases or long unmatched sub-sequences. RCC uses more than one reference sequences in its compression.

Two datasets are considered. The first dataset is s.aureus which contains a genome of bacteria *Staphylococcus aureus* of 17 samples. The second dataset is e.coli which contains 33 genome samples of bacteria *Escherichia coli*. Both datasets can be downloaded at Ensembl (http://ensemblgenomes.org/). Windows system was adopted for implementation. All experiments were conducted on a computer with 2.66 Hz dual core CPU.

Table 3 summarises the compression performance in terms of the bpb. We can see that methods in Group 1 have bpb ranging from 0.51 to 1.66. In contrast, methods in Group 3 range from 0.23 to 0.77. This shows the importance of the selection of reference sequences. In RLCSA, RLZ-opt and iDoComp methods, the user has to select one sequence as the reference. But methods such as GDC and FRESCO enhance the selected reference sequence using extra phrases or rewriting. This has a positive effect on the compression performance.

However, it is always advantageous to use more than one reference sequence. We can see that the use of clustering and multiple reference sequences in RCC significantly outperform all other methods in population sequence compression. This shows that finding sub-structures within a set of population sequences is crucial for the compression performance. The flexibility in defining the set of reference sequences such as using statistical models for predicting the base composition can further be considered in the design of future compression method.

Figs. 7 and 8 show, respectively, the plots of the time taken for compression and decompression against the bpb performance for s.aureus and e.coli. The best algorithm should be the one that has small bpb and short compression/decompression time, i.e. the point locates at the lower left corner of the plot. RCC has the smallest bpb but the highest compression/decompression time. GDC has the second smallest bpb and its computations are significantly lower than RCC. Despite the fact that FRESCO allows reference rewriting while iDoCOMP simply chooses one sequence as the reference, their performances are comparable. In general, methods in Group 3 perform better than methods in Group 1. Note that the

Table 3 Performance of different DNA compression algorithms on two sets of population sequences: s.aureus and e.coli

	Group 1			Group 2		Group 3	
	RLCSA [46] 2010	RLZ-opt [48] 2011	iDoCOMP [51] 2015	MSC [56] 2015	GDC [57, 58] 2011/2015	FRESCO [59] 2013	RCC [60] 2018
bpb for s.aureus	1.45	0.64	**0.51**	0.46	0.27	0.36	**0.23**
bpb for e.coli	1.66	0.91	**0.70**	0.46	0.38	0.77	**0.28**

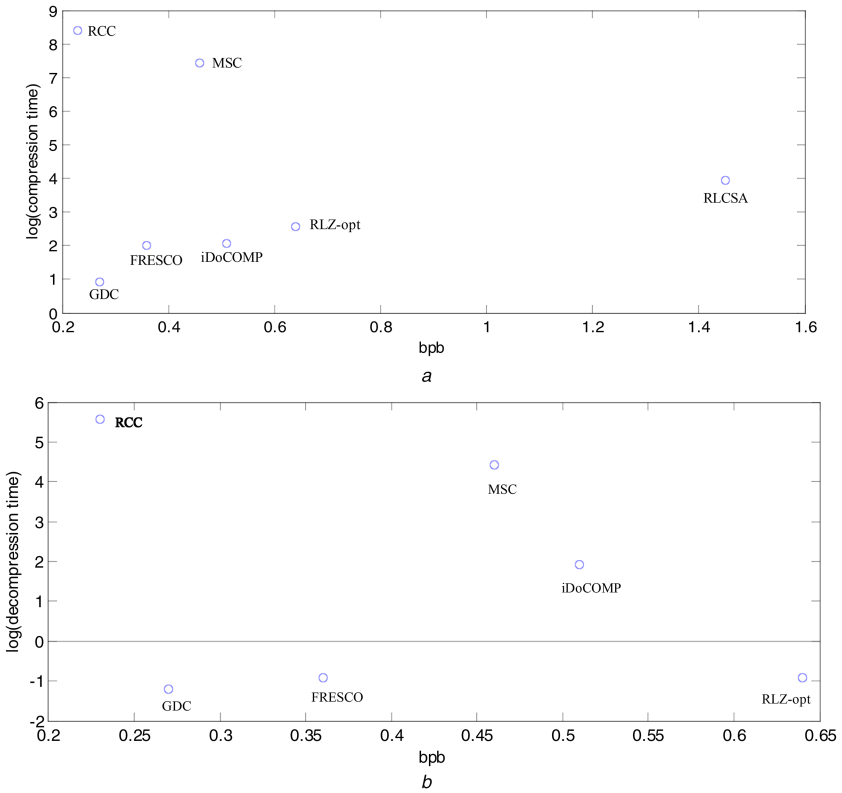

Fig. 7 *Plots of*
(a) Log of the compression time, ***(b)*** Log of the decompression time against bpb for s.aureus

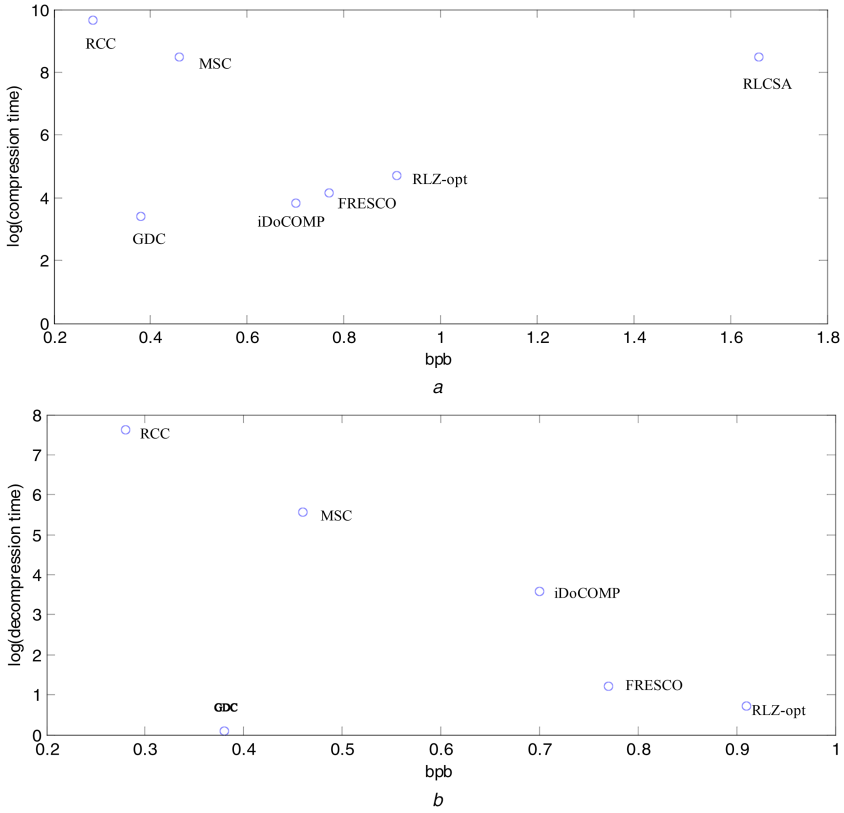

Fig. 8 *Plots of*
(a) Log of the compression time, ***(b)*** Log of the decompression time against bpb for e.coli

Table 4 bpb of different groups of sequences by compressing them separately and by using MSC

	bpb in separate compression	bpb achieved by MSC
S. cerevisiae: 6 chromosomes	1.9226	1.8625
ECA	1.8886	1.2742
ECB	1.9037	0.4461

decompression time is much faster than the compression time for all methods. Hence, in large-scale data distribution where compression is done once in the server, but decompression is performed by the user, all these methods are practical to be applied in the actual data distribution and sharing.

Figs. 7, 8 and Table 3 show that the use of multiple reference sequences can achieve the smallest bpb at the expense of computations. In RCC, normalised histogram is used to characterise a DNA sequence. K-means clustering is then performed on the normalised histogram in sequences clustering. In fact, other features such as k-mer frequencies can be used to characterise a DNA sequence and perform sequence similarity analysis. The combination of k-mer with the wavelet transform provides good characterisation of DNA sequences which can be explored for sequence clustering [83].

Further study is required to investigate the effect of using different combination of features and clustering on the compression performance in terms of both bpb and complexity considerations. Besides, as clustering is performed on the population sequences, the compression time is significantly increased. Various methods to improve the computational complexity can be investigated, such as the use of parallel computing platform [53].

6 Reference-based DNA compression methods for partially similar sequences

Partial similarities between two different DNA sequences are well known [42–44] and have been discussed in Section 2.2. These kinds of similarities are significant for compression because the size of inter-sequence repeats can be twice of the size of intra-sequence repeats [N43]. However, not all the methods in Section 5 can provide effective compression for sequences which only share partial similarity in certain portions. Methods that can be applied for characterising partial similarity include COMRAD [52, 53], ERGC [54], CoGI [55], MSC [56] and referential compression algorithm [81].

These algorithms could be extended to work as reference-based compression for a single sequence. In particular, sub-sequence matches are first performed between the target sequence to be compressed and reference sequences(s). The reference sequence can be the target sequence itself exploring intra-sequence similarity or another sequence exploring inter-sequence similarity. The sub-sequence matches are encoded which are then combined with the code of the residue to form an output bitstream. Both exact and approximate matches are considered by these methods.

The performance of these methods depends on the degree of similarities among sequences in the dataset. Table 4 shows the results of MSC in compressing a group of DNA sequences sharing a different degree of similarities [56]. For the chromosome sequences of *S. cerevisiae*, they have a certain degree of similarities, but the similarities are not too high. Despite that, the bpb still reduces from 1.92 to 1.86 if these chromosome sequences are compressed together rather than separately. ECA contains four DNA sequences of *E. coli* and its related species, while ECB contains six DNA sequences of three different groups of *E. coli*. We can see that the use of partial similarities is able to reduce the bpb significantly.

In fact, the bpb reflects the similarities of the group of DNA sequences to be compressed. ECB contains different groups of the same species, while ECA consists of sequences of related species. Thus, sequences in ECB should be more similar than those in ECA. This is in line with the findings that the bpb in ECB is significantly lower than that in ECA. Indeed, this property enables compression methods to be used for quantifying the relationship among different species. For example, the compression method has been used to build the evolution tree of mammalian DNA sequences [84]. It has also used to discover new sub-families of Alu-sequences [85, 86]. Hence, the main purpose of using compression methods in these

cases are not to reduce the storage size, rather they are used to discover the hidden relationship among the sequences, such as whether they are coming from the same family tree.

7 Conclusion

Advances in biological and sequencing technologies have been providing us an explosive growth in genome data. The large-scale distribution and management of DNA sequences are challenging without a compression framework. In this paper, we have reviewed the properties of DNA sequences that can be exploited for compression. Two classes of DNA sequence compression methods have been discussed. The first class of methods considers similar sub-sequences found within the target DNA sequence to be compressed only. The second class extends the intra-sequence similarities to consider similarities from other sequences as well. It mainly focuses on population sequences generated from the NGS techniques. For each class of method, their basic principles and compression performance have been discussed.

For population sequences compression, one key issue is the choice of the reference sequence. The reference sequence can be selected as one sequence within the population or can be constructed by considering the statistics of all sequences within the population. Besides, the reference sequence can also be modified during progressive compression of sequences in a population dataset. Our experimental results show that by having a strategy to flexibly select, construct and modify the reference sequence, the compression performance can be significantly improved. Besides, the use of more than one reference sequences always improves the compression gain because the sub-structure exist in the population sequences can be characterised by different reference sequences. Further study is, however, required to develop an efficient framework of choosing more than one reference sequence in population sequences. Strategies such as parallel computing can be used to reduce the computational complexity. With these improvements, large-scale DNA sequence storage and distribution should be possible. Besides, other signal processing techniques such as compressive sensing could be explored for DNA sequence compression. For example, in reference-based compression methods for population sequences, the difference between the target sequence to be compressed and the reference sequence can be considered to be a 'sparse' signal if these two sequences are highly similar to each other. In this way, compressive sensing can be applied to the sparse signal to achieve compression. This would be a future direction for further study by the signal processing community.

8 Acknowledgment

The author thanks the reviewers, associate editor and editor for their comments and suggestions which help in improving the overall quality of this article.

9 References

[1] Yang, Y., Xie, B., Yan, J.: 'Application of next-generation sequencing technology in forensic science', *Genomics Proteomics Bioinformatics*, 2014, **12**, pp. 190–197

[2] Weber-Lehmann, J., Schilling, E., Gradl, G., *et al.*: 'Finding the needle in the haystack: differentiating identical twins in paternity testing and forensics by ultra-deep next generation sequencing', *Forensic Sci. Int.: Genet.*, 2014, **9**, pp. 42–46

[3] Rana, A.K.: 'Crime investigation through DNA methylation analysis: methods and applications in forensics', *Egypt J. Forensic Sci.*, 2018, **8**, p. 7

[4] Guttmacher, A.E., Collins, F.S.: 'Realizing the promise of genomics in biomedical research', *J. Am. Med. Assoc.*, 2005, **294**, (11), pp. 1399–1402

[5] Zhang, J., Baran, J., Cros, A., *et al.*: 'International cancer genome consortium data portal – a one-stop shop for cancer genomics data', *Database: J. Biol. Databases Curation*, 2011, **2011**, doi: 10.1093/database/bar026

[6] Li, H.: 'BGT: efficient and flexible genotype query across many samples', *Bioinformatics*, 2016, **32**, (4), pp. 590–592

[7] Celniker, S.E., Dillon, L.A., Gerstein, M.B., *et al.*: 'Unlocking the secrets of the genome', *Nature*, 2009, **459**, (7249), pp. 927–930

[8] Chen, X., Kwong, S., Li, M.: 'Compression algorithm for DNA sequences and its applications in genome comparison'. 4th Annual Int. Conf. on Computational Molecular Biology, Japan, 2000, p. 107

[9] Keogh, E.J., Lonardi, S., Ratanamahatana, C.A.: 'Towards parameter-free data mining'. The 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, Seattle, 2004, pp. 206–215

[10] Pratas, D., Silva, R.M., Pinho, A.J., *et al.*: 'An alignment-free method to find and visualize rearrangements between pairs of DNA sequences', *Sci. Rep.*, 2015, p. 5:10203, doi:10.1038/srep10203

[11] 1000 Genomes. Available at http://www.1000genomes.org/

[12] Christley, S., Lu, Y., Li, C., *et al.*: 'Human genomes as email attachments', *Bioinformatics*, 2009, **25**, (2), pp. 274–275

[13] Benson, D.A., Karsch-Mizrachi, I., Lipman, D.J., *et al.*: 'Genbank', *Nucleic Acids Res.*, 2005, **33**, pp. D34–D38

[14] Brooksbank, C., Cameron, G., Thornton, J.: 'The European bioinformatics institute's data resources', *Nucleic Acids Res.*, 2010, **38**, pp. 17–25

[15] Sugawara, H., Ogasawara, O., Okubo, K., *et al.*: 'DDBJ with new system and face', *Nucleic Acids Res.*, 2008, **36**, pp. D22–D24

[16] Hosseini, M., Pratas, D., Pinho, A.J.: 'A survey on data compression methods for biological sequences', *Information*, 2016, **7**, (56), doi: 10.3390/info7040056

[17] Zhu, Z., Zhang, Y., Ji, Z., *et al.*: 'High-throughput DNA sequence data compression', *Brief. Bioinform.*, 2015, **16**, (1), pp. 1–15. Available at https://doi.org/10.1093/bib/bbt087

[18] Available at http://www.ncbi.nlm.nih.gov/Genbank/

[19] Consortium, I.C.G.: 'International network of cancer genome projects', *Nature*, 2010, **464**, (7291), pp. 993–998. Available at http://dx.doi.org/10.1038/nature08987

[20] Brierley, C.: 'Ten years on, Wellcome trust launches study of 10,000 human genomes in UK', 2010. Available at http://www.wellcome.ac.uk/News/Media-office /Press-releases/2010/WTX060061.htm

[21] Lu, S., Chen, H., Peng, L., *et al.*: 'A compression algorithm of FASTQ file based on distribution characteristics analysis'. 13th Int. Conf. on Computer Science and Education, Sri Lanka, 2018, pp. 1–5

[22] Deorowicz, S., Grabowski, S.: 'Data compression for sequencing data', *Algorithms. Mol. Biol.*, 2013, **8**, p. 25

[23] Grumbach, S., Tahi, F.: 'A new challenge for compression algorithms: genetic sequences', *J. Inf. Process. Inf. Manage.*, 1994, **30**, (6), pp. 875–886

[24] Grumbach, S., Tahi, F.: 'Compression of DNA sequences'. Data Compression Conf., Snowbird, 1993, pp. 340–350

[25] Rivals, E., Delahaye, J.-P., Dauchet, M., *et al.*: 'A guaranteed compression scheme for repetitive DNA sequences'. Data Compression Conf., Snowbird, 1996, p. 453

[26] Chen, X., Kwong, S., Li, M.: 'A compression algorithm for DNA sequences', *IEEE Eng. Med. Biol. Mag.*, 2001, **20**, (4), pp. 61–66

[27] Chen, X., Kwong, S., Li, M.: 'A compression algorithm for DNA sequences and its applications in genome comparison', *Genome Inform.*, 1999, **10**, pp. 51–61

[28] Li, M., Badger, J.H., Chen, X., *et al.*: 'An information-based sequence distance and its application to whole mitochondrial genome phylogeny', *Bioinformatics*, 2001, **17**, (2), pp. 149–154

[29] Matsumoto, T., Sadakane, K., Imai, H.: 'Biological sequence compression algorithms', *Genome Inform.*, 2000, **11**, pp. 43–52

[30] Chen, X., Li, M., Ma, B., *et al.*: 'DNACompress: fast and effective DNA sequence compression', *Bioinformatics*, 2002, **18**, (12), pp. 1696–1698

[31] Cherniavsky, N., Ladner, R.: 'Grammar-based compression of DNA sequences'. UW CSE Technical Report (2007–05–02), 2004. Available at https://personal.broadinstitute.org/neva/publications/dnasequitur.pdf

[32] Tabus, I., Korodi, G., Rissanen, J.: 'DNA sequence compression using the normalized maximum likelihood model for discrete regression'. Data Compression Conf., USA, 2003, pp. 253–262

[33] Korodi, G., Tabus, I.: 'An efficient normalized maximum likelihood algorithm for DNA sequence compression', *ACM Trans. Inf. Syst.*, 2005, **23**, (1), pp. 3–34

[34] Cao, M.D., Dix, T.I., Allison, L., *et al.*: 'A simple statistical algorithm for biological sequence compression'. Data Compression Conf., Snowbird, 2007, pp. 43–52

[35] Rajeswari, P.R., Apparao, A.: 'Genbit compress tool (GBC): a Java-based tool to compress DNA sequences and compute compression ratio (BITS/BASE) of genomes', *Int. J. Comput. Sci. Inf. Technol.*, 2010, **2**, (3), pp. 181–191

[36] Rajeswari, P.R., Apparao, A., Kumar, R.K.: 'HUFFBIT COMPRESS – algorithm to compress DNA sequences using extended binary trees', *J. Theoretical Appl. Inf. Technol.*, 2010, **13**, pp. 101–106

[37] Rajeswari, P.R., Apparao, A.: 'DNABIT compress – genome compression algorithm', *Bioinformation*, 2011, **5**, (8), pp. 350–360

[38] Panneer Arokiaraj, S., Robert, L.: 'Pattern recognition based DNA sequence compressor'. 2012 IEEE Int. Conf. on Computational Intelligence and Computing Research, India, 2012, pp. 1–5

[39] Sardaraz, M., Tahir, M., Ikram, A.A., *et al.*: 'Seqcompress: an algorithm for biological sequence compression', *Genomics*, 2014, **104**, (4), pp. 224–228

[40] Saada, B., Zhang, J.: 'DNA sequence compression technique based on modified DNABIT algorithm'. Proc. of the World Congress on Engineering, London, 2016, vol I

[41] Al-Okaily, A., Almarri, B., Al Yami, S., *et al.*: 'Toward a better compression for DNA sequences using Huffman encoding', *J. Comput. Biol.*, 2017, **24**, (4), pp. 280–288

[42] Wu, C.-P.P., Law, N.F., Siu, W.C.: 'Cross chromosomal similarity for DNA sequence compression', *Bioinformation*, 2008, **2**, (9), pp. 412–416

[43] Wu, P., Law, N.F., Siu, W.C.: 'Analysis of cross sequence similarities for multiple DNA sequences compression', *Int. J. Comput. Aided Eng. Technol.*, 2009, **1**, (4), pp. 437–454

[44] Hanus, P., Dingel, J., Chalkidis, G., *et al.*: 'Compression of whole genome alignments', *IEEE Trans. Inf. Theory*, 2010, **56**, (2), pp. 696–705

[45] Brandon, M.C., Wallace, D.C., Baldi, P.: 'Data structures and compression algorithms for genomic sequence data', *Bioinformatics*, 2009, **25**, (14), pp. 1731–1738
[46] Makinen, V., Navarro, G., Siren, J., *et al.*: 'Storage and retrieval of highly repetitive sequence collections', *J. Comput. Biol.*, 2010, **17**, (3), pp. 281–308
[47] Kuruppu, S., Puglisi, S.J., Zobel, J.: 'Relative Lempel-Ziv compression of genomes for large-scale storage and retrieval'. Int. Symp. on String Processing and Information Retrieval, Mexico, 2010, vol. 6393, pp. 201–206
[48] Kuruppu, S., Puglisi, S.J., Zobel, J.: 'Optimized relative Lempel-Ziv compression of genomes'. Proc. of 34th Australasian Computer Science Conf., Perth, 2011, vol. 113, pp. 91–98
[49] Wang, C., Zhang, D.: 'A novel compression tool for efficient storage of genome resequencing data', *Nucleic Acids Res.*, 2011, **39**, (7), doi: 10.1093/nar/gkr009
[50] Pinho, A.J., Pratas, D., Garcia, S.P.: 'GReen: a tool for efficient compression of genome resequencing data', *Nucleic Acids Res.*, 2012, **40**, (4), doi: 10.1093/nar/gkr1124
[51] Ochoa, I., Hernaez, M., Weissman, T.: 'Idocomp: a compression scheme for assembled genomes', *Bioinformatics*, 2015, **31**, (5), pp. 626–633
[52] Kuruppu, S., Beresford-Smith, B., Conway, T., *et al.*: 'Iterative dictionary construction for compression of large DNA data sets', *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 2012, **9**, (1), pp. 137–149
[53] Biji, C.L., Madhu, M.K., Vishnu, V., *et al.*: 'Compression of large genome datasets using COMRAD on parallel computing platform', *Bioinformation*, 2015, **11**, (5), pp. 267–271
[54] Saha, S., Rajasekaran, A.: 'ERGC: an efficient referential genome compression algorithm', *Bioinformatics*, 2015, **31**, (21), pp. 3468–3475
[55] Xie, X., Zhou, S., Guan, J.: 'CoGI: towards compressing genomes as an image', *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 2015, **12**, (6), pp. 1275–1285
[56] Cheng, K.O., Wu, P., Law, N.F., *et al.*: 'Compression of multiple DNA sequences using intra-sequence and inter-sequence similarities', *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 2015, **12**, (6), pp. 1322–1332
[57] Deorowicz, S., Grabowski, S.: 'Robust relative compression of genomes with random access', *Bioinformatics*, 2011, **27**, (21), pp. 2979–2986
[58] Deorowicz, S., Danek, A., Niemiec, M.: 'GDC 2: compression of large collections of genomes', *Sci. Rep.*, 2015, **5**, p. 11565. Available at https://www.nature.com/articles/srep11565
[59] Wandelt, S., Leser, U.: 'FRESCO: referential compression of highly similar sequences', *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 2013, **10**, (5), pp. 1275–1288
[60] Cheng, K.-O., Law, N.-F., Siu, W.-C.: 'Clustering-based compression for population DNA sequences', *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 2019, **16**, (1), pp. 208–211
[61] Jahaan, A., Ravi, D.R.T.N., Panneer Arokiaraj, S.: 'A comparative study and survey on existing DNA compression techniques', *Int. J. Adv. Res. Comput. Sci.*, 2017, **8**, (3), pp. 732–735
[62] Bakr, N.S., Sharawi, A.A.: 'DNA lossless compression algorithms: review', *Am. J. Bioinformat. Res.*, 2013, **3**, (3), pp. 72–81
[63] Cheng, K.O., Law, N.F.: 'A survey of techniques for sequence similarities matching in compression', *Adv. Robot. Autom.*, 2014, **3**, p. 118
[64] Jason de Koning, A.P., Gu, W., Castoe, T.A., *et al.*: 'Repetitive elements may comprise over two-thirds of the human genome', *PLoS Genet.*, 2011, **7**, (12), p. e1002384
[65] Deininger, P.: 'Alu elements: know the SINEs', *Genome Biol.*, 2011, **12**, p. 236
[66] Pascale, E., Liu, C., Valle, E., *et al.*: 'The evolution of long interspersed repeated DNA (L1, LINE 1) as revealed by the analysis of an ancient rodent L1 DNA family', *J. Mol. Evol.*, 1993, **36**, (1), pp. 9–20
[67] Ziv, J., Lempel, A.: 'A universal algorithm for sequential data compression', *IEEE Trans. Inf. Theory*, 1977, **23**, (3), pp. 337–343
[68] Overview of the tool BLAST. Available at http://www.ncbi.nlm.nih.gov/blast/blast_overview.shtml
[69] Ma, B., Tromp, J., Li, M.: 'Patternhunter – faster and more sensitive homology search', *Bioinformatics*, 2002, **18**, pp. 440–445
[70] BLAST. Available at http://blast.genome.jp/
[71] Priyanka Goel, S.: 'A compression algorithm for DNA that uses ASCII values'. IEEE Int. Advance Computing Conf., India, 2014, pp. 739–743
[72] Tan, L., Sun, J.: 'K-means clustering based compression algorithm for the high-throughput DNA sequence'. Int. Conf. on Audio, Language and Image Processing, Shanghai, 2014
[73] Eric, P.V., Gopalakrishnan, G., Karunakaran, M.: 'An optimal seed based compression algorithm for DNA sequences', *Adv. Bioinformatics*, 2016, **2016**. Available at http://dx.doi.org/10.1155/2016/3528406
[74] Challa, R., Pranayani Devi, G., Arava, K., *et al.*: 'A novel compression technique for DNA sequence compaction'. IEEE Int. Conf. on Signal Processing, Communication, Power and Embedded System, India, 2016, pp. 1351–1354
[75] Chen, M., Shao, J.J., Jia, X.M.: 'Genome sequence compression based on optimized context weighting', *Genet. Mol. Res.*, 2017, **16**, (2), doi: 10.4238/gmr16026784
[76] Saada, B., Zhang, J.: 'DNA sequence compression technique based on nucleotides occurrence'. Int. Multi-Conf. of Engineers and Computer Scientists, Hong Kong, 2018, vol. I
[77] Habib, N., Ahmed, K., Jabin, I., *et al.*: 'Modified HuffBit compress algorithm – an application of R', *J. Integr. Bioinform.*, 2018, **15**, (3), doi: 10/1515/job-2017-0057
[78] Cai, N., Bigdeli, T., Kretzschmar, W., *et al.*: 'Sparse whole-genome sequencing identifies two loci for major depressive disorder', *Nature*, 2015, **523**, (7562), pp. 588–591
[79] Sudmant, P.H., Rausch, T., Gardner, E.J., *et al.*: 'An integrated map of structural variation in 2,504 human genomes', *Nature*, 2015, **526**, (7571), pp. 75–81
[80] Andrews, R.M., Kubacka, L., Chinnery, P.F., *et al.*: 'Reanalysis and revision of the Cambridge reference sequence for human mitochondrial DNA', *Nat Genet.*, 1999, **23**, (2), p. 147
[81] Mehta, K., Ghrera, S.P.: 'DNA compression using referential compression algorithm'. IEEE Int. Conf. on Contemporary Computing, India, 2015, pp. 64–69
[82] Banerjee, K., Prasad, R.A.: 'Reference based inter chromosomal similarity based DNA sequence compression algorithm'. IEEE Int. Conf. on Computing, Communication and Automation, India, 2017, pp. 234–238
[83] Lin, J., Wei, J., Adjeroh, D., *et al.*: 'SSAW: a new sequence similarity analysis method based on the stationary discrete wavelet transform', *BMC Bioinformatics*, 2018, **19**, p. 165
[84] Li, M., Badger, J.H., Chen, X., *et al.*: 'An information based distance and its application to whole mitochondrial genome phylogeny', *Bioinformatics*, 2001, **17**, (2), pp. 149–154
[85] Allison, L., Stern, L., Edgoose, T., *et al.*: 'Sequence complexity for biological sequence analysis', *Comput. Chem.*, 2000, **24**, pp. 43–55
[86] Milosavljevic, A., Jurka, J.: 'Discovery by minimal length encoding: A case study in molecular evolution', *Mach. Learn.*, 1993, **12**, (1–3), pp. 68–87