A SURVEY ON DNA SEQUENCE COMPRESSION ALGORITHMS

Arunachalaprabu G. [1], Fathima Bibi K. [2]
[1]Research Scholar in Computer Science, Thanthai Periyar Govt. Arts & Science College (A), Tiruchirappalli
E-mail: guruarun12@gmail.com
[2]Assistant Professor in Computer Science, Thanthai Periyar Govt. Arts & Science College (A), Tiruchirappalli
E-mail: kfatima72@gmail.com

Abstract. *Deoxyribonucleic Acid (DNA) plays a major role in the development, growth and reproduction of all living organisms. Due to the recent development of scientific researches in biology, virology and medicine public databases are over flooded with enormous amount of DNA data. It not only faces severe challenges like storage but also restricts transmission capacity and retrieval process. Lossless DNA Compression is used to reduce the size of data, improve the capacity of storage medium and henceforth vast amount of data can be transmitted at any given time. There are many existing lossless DNA compression algorithms most of them of which are not suitable for compressing the DNA data. In addition, the development of compression algorithms that help to reduce the size of DNA data is rather a difficult task. This paper discusses the recent researches on various lossless compression algorithms. Reviews on standard algorithms are briefed. The study shows that compression of DNA sequence is vital for understanding the essential characteristics of DNA data. Two major categories namely, horizontal mode and vertical mode are focused. A comparative study about the notions of the different modes of DNA compression algorithms is analysed. To evaluate the performance of DNA compression algorithms commonly used metrics such as compression ratio, saving percentage and time taken for compression and decompression were used. An outline of some research problems that assist for further development of effective compression algorithms for DNA data and the scope for future enhancement are also discussed.*

Keywords: *Bioinformatics, Deoxyribonucleic Acid, Horizontal mode, Vertical mode, Compression Ratio.*

1. INTRODUCTION

Bioinformatics is a broad multi-disciplinary field that aims to solve biological problems using Deoxyribonucleic Acid and other related information. Deoxyribonucleic Acid, or DNA, is a long, linear vital molecule of living organisms. The primary structure of DNA molecule is a double helix strand made up of four molecules or bases namely, Adenine (A), Cytosine (C), Guanine (G), and Thymine (T).

A DNA sequence is an elongated string which comprises a set of consecutive bases (Example: chmpxx sequence - TTGAACGAGAAGCCGTATGAAATGAAAATAT). Many researches in bioinformatics focus on the study of DNA sequences based on their functions and features. For instance, diseased DNA sequences are compared with healthiest ones to detect the major differences between them. Besides, the DNA sequences are analyzed to identify similarity between patterns. For these reasons, huge amount of DNA sequences are stored in databases. When the length of the DNA sequence increase rapidly, storage and transmission become significantly harder. In addition, it causes a major issue for many analysis tasks owing to its high memory usage and cost for computation.

Compression is an effective way for reducing the size of DNA sequence. The basic concept behind compression is to reduce the number of bits needed to store DNA sequences as they can lead to improved storage capacity and minimum network traffic. The need for compression algorithms and expertise has increased as Genome Projects resulted in an exponential growth in DNA databases. With years of research and development, there are several DNA compression algorithms available to reduce the size of DNA sequence. Compression algorithms are primarily of two types: Lossy and lossless.

- Lossy involves loss of information.
- Lossless results in no loss of information.

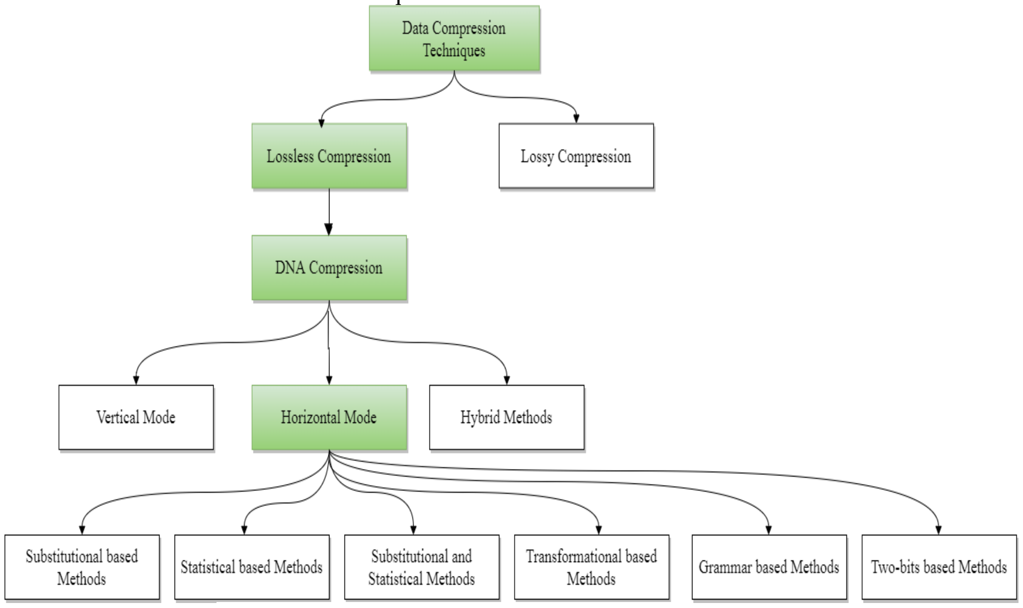
There are many situations that require compression where the reconstruction is to be identical to the original. In addition, there are also numerous situations in which it is not possible to relax this requirement. This opens a challenging question in research fields, such as how to reduce the size of DNA sequence without sacrificing loss of information. Therefore, lossless compression algorithms that best approximate the original dataset with reduced storage cost are likely to play an important role in DNA sequence compression.

The paper presents a general study of DNA compression algorithms that have been useful to reduce the length of the DNA sequences. Most text compression algorithms have focused on the compression of DNA sequences. However, DNA sequences often consist of many repeated and non-repeated bases. It is not easy to compress DNA sequence with good compression ratio using text compression algorithms. Some interesting compression algorithms include LZ77 (Ziv and Lempel, 77), LZ78, Prediction with Partial Match (PPM), Context Tree Weighting (CTW), GNU zip (GZip), Compress method and Bzip2. LZ77 retains a dictionary in which previously encoded input stream is stored. Sliding window method is used to examine the input stream. It is divided into two buffers: 1) Search buffer – holds recently encoded stream and 2) Look-ahead buffer – holds next segment of the stream to be encoded. At the decoding phase, a buffer is maintained equal in size to the encoder's window. A good compression ratio is achieved for many sequences. Though it requires less amount of memory more time was taken to encode the sequences [1]. LZ78 (Ziv and Lempel, 1978) uses dictionary for both encoder and decoder instead of any search buffer, look-ahead buffer or sliding window [2]. PPM method (Cleary and Witten, 1984) compresses the DNA sequences with compression ratio greater than two bits per base (bpb) [3]. CTW (Willems et al., 1995) is suitable to compress the DNA sequences below 2 bpb [4]. GZip (Jean-loup Gailly and Mark Adler, 1992) uses adaptive Lempel-Ziv coding to compress the named files in deflate mode [5]. The performance of Compress method (Terry Welch, 1984) based on LZW coding is high with minimum memory requirements. Nevertheless, the compression ratio of compress method is significantly low [6]. In Bzip2 (Julian Seward, 1996), Burrows-Wheeler block sorting technique and Huffman coding are used to reduce the size of files [7]. However, most traditional compression algorithms have not achieved good compression results.

The paper is organized as follows: Section 2 categorizes the different DNA sequence compression techniques. The formulae of the commonly used performance metrics are shown in Section 3. Section 4 describes the recent horizontal mode DNA sequence compression algorithms. Reviews of vertical mode DNA sequence compression algorithms are discussed in Section 5. Experimental results of hybrid algorithms are shown in Section 6. Finally, Section 7 summarizes the different lossless DNA sequence compression algorithms.

2. TAXONOMY OF DNA SEQUENCE COMPRESSION TECHNIQUES

This section gives an overview of the techniques reviewed in DNA sequence compression algorithms. The classification of different DNA sequence compression algorithms are shown in Figure 1. DNA compression algorithms are classically split into two common methods: Horizontal mode and Vertical mode.


Data Compression Techniques
Lossless Compression
Lossy Compression
DNA Compression
Vertical Mode
Horizontal Mode
Hybrid Methods
Substitutional based Methods
Statistical based Methods
Substitutional and Statistical Methods
Transformational based Methods
Grammar based Methods
Two-bits based Methods
Figure 1: Taxonomy of DNA Compression Techniques

2.1 HORIZONTAL MODE

The horizontal mode compresses a sequence based on its information i.e., sequences are compressed successively. Broadly speaking, horizontal mode compression algorithms are divided into the following categories:

- Substitutional based methods – A dictionary of frequently appearing bases is maintained and when these bases appear in the sequence they are replaced by the codeword in dictionary.
- Statistical based methods – Variable size short codes are assigned to frequently appearing bases or set of bases in the sequence.
- Substitutional and Statistical based methods – Features of both substitutional and statistical methods are used to encode the sequence.
- Transformational based methods – Transformations takes place in the actual sequence and compression is applied only on the transformed sequence.
- Grammar based methods – Compresses a text string using context-free grammar. The compressed string is encoded by a symbol which in turn is converted to binary [8].
- Two-bit based methods – Unique binary bits are assigned for the bases (A = 00, C = 01, G = 10, and T = 11).

2.2 VERTICAL MODE

The vertical mode works by using the information stuck between two sequences by referring to the information contained in only one of the sequence.

3. PERFORMANCE METRICS

The effectiveness of a compression algorithm can be evaluated in various ways:

3.1 COMPRESSION RATIO (CR)

The compression ratio is the ratio between compressed file size and original file size. Compression ratio is formally expressed in bpb or bits per character (bpc).

CR = Compressed file size / Original file size

3.2 COMPRESSION FACTOR (CF)

The compression factor is the ratio between original file size and compressed file size. Compression factor is the inverse of compression ratio.

CF = Original file size / Compressed file size

3.3 SAVING PERCENTAGE (SP)

Saving percentage is the difference between original file size and compressed file size to the size of original file.

SP = (Original file size -Compressed file size) / Original file size

3.4 COMPRESSION TIME

Compression time refers to the amount of time, in milliseconds, needed to compress the file.

3.5 DECOMPRESSION TIME

Decompression time refers to the time required to decompress the compressed file to its original form. Decompression time is expressed in milliseconds.

4. HORIZONTAL MODE ALGORITHMS

With sophisticated DNA compression tasks, there is much opportunity for research and development of advanced, effectual, and scalable horizontal mode DNA compression methods in bio-informatics. Some interesting methods are:

4.1 SUBSTITUTIONAL BASED METHODS

Most compression algorithms are based on substitutional based methods. Murugan and Punitha, (2021) have designed a Pattern Matching Extended Compression Algorithm (PMECA) to compress the DNA sequence. PMECA is the extension of improved-compress algorithm [9]. First, it scans segments of the sequence and identifies identical patterns. Based on the number of bases, the patterns are stored in dictionary either in permanent or temporary manner. Matchless patterns are converted and grouped into zeros and ones. Standard datasets taken from GenBank of National Center for Biotechnology Information (NCBI) [10] was used for analysis. The algorithm resulted with a compression ratio of 91%. Simulation results have shown significant improvement of speed and reduction in file size over existing algorithms [11].

Cui et al., (2020) proposed a new approach using deep learning and arithmetic coding. In the preprocessing step, sliding window of the sequence was transformed into vectors. The local and global features are mined using Convolutional Neural Network (CNN) and Bi-directional Long Short-Term Memory Networks (BiLSTM) model. The algorithm is 3.7 times better compared to DeepDNA [12].

GeCo2 tool is an enhanced version of GeCo tool developed by Pratas et al., (2020) [13]. The genomic sequences compressed using this method are combined with cache-hash sizes, inverted repeats, interface for command line, novel pre-computed levels, and different code optimizations. The algorithm resulted with 0.2142% saving percentage when compared with GeCo.

Hui Chen (2020) suggested a genome sequence compression algorithm using entropy coding technique based on context modeling. The sequences are divided and transformed into four clusters, namely, coding sequence cluster, intron cluster, RNA cluster and residual cluster. Each set will be arranged corresponding to certain characteristics of the sequences which are encoded using entropy coding technique. The method was tested with benchmark datasets taken from US Genbank database. The algorithm resulted with an average compression ratio of 1.72 bpb [14].

Mansouri et al., (2020) described a novel lossless DNA Compression Algorithm based on Single-Block Encoding Scheme (DNAC-SBE). There are three phases namely, i) One-Bit Method Phase – position of bases with high frequencies is replaced by ones and others by zeros. ii) Single-Block Encoding Phase – encodes the generated streams and iii) Third Phase – assigns shortest codeword for each block dynamically. It is observed that DNAC-SBE has outperformed the other DNA sequence compression algorithms [15].

Shan E Zahra et al., (2019) [16] presented the Run Length Index Based Coding (RLIBC) algorithm. The basic steps are: 1) Remove all redundant DNA sequence from the input genomic dataset and store its index number 2) Perform segmentation process on each segment 3) Finally, compare each segment with index and transform the index number into binary code. When compared with other algorithms RLIBC has achieved an average compression ratio of 1.75 bpb and average compression factor of 5.7311. Data savings is 82.6% and average time taken for compression and decompression is one second.

Ayad E. Korial and Ali Kamal Taqi (2018) proposed a novel technique A2 to reduce the file size. The algorithm consists of four stages to make the substitutional model. The first stage is a modified version of Run Length Encoding which generates a symbol. The next two stages perform pre-mapping and post-mapping and the final stage develops a permutation technique using Burrows-Wheeler Transform (BWT) method. The algorithm achieved better compression ratio and saving storage space when compared with GenCompress [17]. The results of the various substitutional based compression methods are given in Table 1.

Table 1: Performance Evaluation of Substitutional Based Compression Methods

Methodology	**Dataset**	**Performance Metrics**		**Drawback**
		Saving Percentage	**CT(sec)**	
PMECA[11]	Humhdystrop	93	1.1	

	Humghcsa	90	2.1				
	Humhbb	90	2.4				
	Humhdabcd	91	1.9				
	Humhprtb	90	1.6				
Deep Learning Model[12]		CR(bpb)					
	Fish	0.70					
	Birds	0.66					
	Human	0.01					
	Ray-finned fishes	0.81					
GeCo2[13]		No. of bytes needed	CT(sec)				
	HoSa	38845642	652.4				
	GaGa	33877671	494.7				
	DaRe	11488819	198.8				
	OrSa	8646543	138.3				
	DrMe	7481093	102.4				
	EnIn	5170889	82.5				
	ScPo	2518963	34.2				
	PlFa	1925726	35.3				
	EsCo	1098552	5.1				
	HaHi	902831	4.4				
	AeCa	380115	1.9				
	HePy	375481	1.9				
Entropy Coding Technique[14]		CR(bpb)	CT(sec)				High compression time
	Chmpxx	1.5788	2.06				
	Chntxx	1.5891	1.56				
	Humhprtb	1.8532	0.57				
	Humhbb	1.8318	0.92				
	Vaccg	1.7788	3.13				
DNAC-SBE[15]		CR(bpb)					Does not accept any other data
	Chmpxx	1.604					
	Humhstrop	1.724					
	Humhbb	1.717					
	Humhcvcg	1.741					
	Mpomtcg	1.721					
	Vaccg	1.671					
	Mtpacga	1.650					
	Humhrtb	1.720					
DNAC-SBE[15]		Percentage of Reduction					Does not accept any other data
	NC_017526	23.61					
	NC_017652	22.06					
	Ce10	28.98					
	sacCer3	25.53					
	Eukaryotic	19.71					
DNAC-SBE[15]		C-Size	C-Mem	D-Mem	CT	DT	Does not accept any other data
	NC_017526	0.55	36.71	64.31	1.78	1.24	
	NC_017652	01.06	49.03	101.45	1.67	1.27	
	sacCer3	02.48	5.09	13.54	26.57	9.71	
	Ce10	19.75	68.75	212.75	73.13	45.04	
	Chimpanzee	570.73	522.23	729.9	1445.15	1155.97	
	Korea2009024	577.23	649.56	703.69	1479.57	1003.27	
	Eukaryotic	423.32	1859.00	1349.05	1117.44	903.1	
RLIBC[16]		CR(bpb)	Reduction to %				
	Humhstrop	1.400258	82.49678				

	Humhdabcd	1.414107	82.32366	
	Humhbb	1.409723	82.37846	
	Mpomtcg	1.369194	82.88507	
	Vaccg	1.406292	82.42147	
		Compression Size	Speed	CT(sec)
A2[17]	gbbct45	20.9	0.453 MB/sec	196
	gbbct108	19.1	0.455 MB/sec	195
	gbvrt9	1.70	0.456 MB/sec	17

4.2 STATISTICAL BASED METHODS

Statistical based compression methods are much familiar methods for reducing DNA sequences. Gede Eka Sulistyawan et al., (2020) have suggested a compression system which combines Burrows Wheeler Transform and Hidden Markov Model namely BWT-HMM. BWT was applied to restructure the DNA data which generates numerous redundant bases. The DNA data are segmented according to a single DNA base repeat. Re-estimation algorithm was used to reduce the storage space. The methodology was tested with DNA datasets taken from NCBI. Performance metrics such as compression ratio and time taken for computation were calculated. The proposed algorithm resulted with 4.276 bpb compression ratio with an improved mean compression ratio of 4.004 [18].

Sebastian Deorowicz (2020) introduced FQSqueezer that utilizes partial matching and dynamic Markov coder algorithms for genomic data compression. Experimentation results (Table 2) have shown that this algorithm has achieved better compression ratio for standard benchmark datasets [19].

Table 2: Performance Evaluation of Statistical Based Compression Methods

Methodology	**Dataset**	**Performance Metrics**				**Drawback**
		C-RAM	**D-RAM**	**CT (sec)**	**DT (sec)**	
FQSqueezer [19]	ERR174310_1	91.6	90.6	12728	13100	High memory usage and time
	ERR532393_1	16.4	16.4	1344	1452	
	SRR327342_1	6.7	6.6	144	145	
	SRR554369_1	6.2	6.2	68	70	
	SRR635193_1	12.1	12.1	456	462	
	SRR689233_1	11.7	11.6	406	413	
	SRR870667_1	36.4	36.1	4127	4432	
	SRR1265495_1	13.2	13.1	658	685	
	SRR1265496_1	13.0	13.0	609	652	

4.3 SUBSTITUTIONAL AND STATISTICAL BASED METHODS

It is a hybrid method that combines both substitutional and statistical approaches. Word based compression technique (Sanjeev Kumar et al., (2020)) compresses the genomic data using Modified Word Based Tag Code (MWBTC) and Delta Coding. Tests were conducted using FNA, FEN, Camera, and Eukaryotic datasets. The proposed algorithm helps to search DNA sequence devoid of decompression. When compared to LZMA and Seqcompress more than 20% to 30% better results were obtained (Table 3) [20].

Table 3: Performance Evaluation of Substitutional and Statistical Based Compression Methods

Methodology	**Dataset**	**Performance Metrics**					**Drawback**
		PCR	**C-Memory**	**D-Memory**	**CT**	**DT**	
WBCT[20]	FNA	21.52	356.89	50.78	1593	1357	Memory usage of LZMA is less compared to WBTC
	FEN	20.03	351.25	30.21	1321	1087	
	Camera	10.02	98.59	24.62	786	627	
	Eukaryotic	19.76	99.33	23.69	649	574	

4. 4 TRANSFORMATIONAL BASED METHODS

In transformational methods the DNA sequence is transformed to a specific form before compression to attain good compression ratio. Raju Bhukya (2019) developed a Differential Direct (2D) coding method based on dynamic dictionary approach. The approach works on triplets of DNA sequence bases and patterns of length multiples of three. The dictionary table of 2D coding bifurcates into two parts: i) Static part and ii) Dynamic part. The performance of the algorithm when compared with existing 2D algorithm [21] gave minimum compression ratio with reduced computational time [22].

Jothi et al., (2018) described a lossless segment compression algorithm using Lempel-Ziv Welch technique to reduce the size of DNA sequences. The architecture consists of four parts: a) Upload the DNA sequences b)

Organize the sequences c) Check relationship between two random sequences d) Compress the sequences using LZW technique. The proposed algorithm resulted with an improved compression ratio when compared to Extended ASCII algorithm, Modified RLE algorithm and COMRAD. Experimental results have shown that huge amount of time is required to arrange the sequences [23].

Shengwang Du et al., (2020) designed a compression method where the bases are converted to standard characters in first phase. The characters are compressed using LZ77 algorithm in the subsequent phase. Ten genomes of size 1 to 15M taken from NCBI database were used for testing. The performance of the proposed algorithm was measured using standard metrics such as compression ratio, compression time and decompression time. The time taken for compression and decompression is 83% and 54% respectively [24]. Table 4 gives the performance evaluation of the reviewed transformational based compression methods.

Table 4: Performance Evaluation of Transformational Based Compression Methods

Methodology	**Dataset**	**Performance Metrics**				**Drawback**
		Compressed File Size	**CR**	**CT (sec)**	**DT (sec)**	
Differential Direct Coding (2D) based on Dynamic Dictionary Approach[22]	Bacillus Subtilis	1376213	3.1061	64631	34741	Compression time is high than 2D algorithm
	Escherichia Coil K12 MG1655	1513218	3.1098	70646	37372	
	Mycoplasm Genitalium G37	185424	3.1730	8845	4267	
A compression method for DNA[24]		CR	CT (sec)	DT (sec)		Average Decompression time is minimized by 54%
	NC_017526	75.00	6.004	5.311		
	NC_002942	75.02	5.351	4.947		
	NZ_CP015934	75.05	5.985	5.073		
	NZ_CP015935	75.02	5.529	5.733		
	NZ_CP015938	75.07	5.133	5.060		
	NC_013929	75.17	9.018	17.929		
	NC_014318	75.15	9.870	15.742		
	NC_010162	75.06	12.564	25.590		

4.5 GRAMMAR BASED METHODS

In grammar based methods, context-free grammar is applied on DNA sequences. The grammars are transformed into a set of symbols and finally encoded into binary form. Diego Diaz-Dominguez and Gonzalo Navarro (2020) [25] suggested a grammar based algorithm for collection of reads to construct BWT. The collection of reads is stored as grammar to compute BWT with the support of self-indexes. The method resulted with an average compression ratio of 4.83 bpb. The study have shown that the proposed algorithm outperformed other results such as Big Repair [26], Full-text index in Minutes Space (FM-index) [27] and Run-Length FM (RLFM) [28].

4.6 TWO-BIT BASED METHODS

In two-bit based methods the bases A, C, G and T are encoded by four distinct two-bit binary values 00, 01, 10 and 11. Murugesan (2020) described a novel Codon based compression algorithm [29] based on two bit binary substitution technique. Additional dictionary is not employed to compress or decompress the genome sequence and hence additional memory is not required. Experimental results (Table 5) have shown an average compression ratio of 1.59 bpb with an average compression time of 0.18 seconds.

Table 5: Performance Evaluation of Transformational Based Compression Methods

Methodology	**Dataset**	**Performance Metrics**		**Drawback**
		CR	**CT(sec)**	
Codon Based (proposed) [29]	Humhstrop	1.55	0.095	
	Humhprtb	1.54	0.115	
	Humhbb	1.55	0.156	
	Mpomtcg	1.55	0.281	
	Vaccg	1.57	0.297	

5. VERTICAL MODE ALGORITHMS

This section reviews works that has focused on lossless DNA sequence compression algorithms based on vertical mode (Table 6). Bruno Carpentieri (2020) described a next generation sequencing data compression algorithm [30] to encode the DNA sequence using two bit encoding technique. The algorithm was tested using six DNA files namely, Lambda Virus (48,502 bytes), Homo sapiens.GRCh38.dna (3,072,712,323 bytes), SRR741411_2 (7,982,945,875 bytes), Mais (2,104,355,422 bytes), Cricetus (2,320,022,665 bytes), Pinus (20,547,720,415 bytes) to methodically demonstrate the performance of the algorithm. The results of the proposed algorithm outperformed zip, gzip, and bzip2 algorithms.

Aníbal Guerra et al., (2020) presented UdeACompress, a referential compression algorithm to reduce the size of FASTQ files. The proposed algorithm works as follows: i) First, align the sequences to detect the most appropriate read sequence ii) Next, sort the sequence using radix sort iii) In the third phase, the sequences are encoded using binary map and instruction array techniques iv) Finally, the encoded data and unmapped reads are compressed by low level compression. The variation in file size was 14% smaller compared to the original file. Experimental results show that the time taken for execution and amount of storage was dramatically reduced and the performance of processor was improved [31].

Table 6: Performance Evaluation of Vertical Based Compression Methods

Methodology	**Dataset**	**Performance Metrics**					**Drawback**
		CR					
Proposed Algorithm [30]	LambdaVirus.fa	3.97					High Computational cost
	Homo_sapiens.GRCh38dna_sm	4.11					
	SRR741411_2	4.02					
	Mais	3.91					
	Cricetus	4.00					
	Pinus	4.01					
		CR	CT	DT	Peak memory consumption		
					C	D	
UdeACompress[31]	SRR1282409	7.29	2.8	10.9	10639	9691	High memory usage and CPU requirements. Speed is sensitive.
	SRR3141946	6.6	3.0	11.5	7578	7030	
	DRR000604	8	2.7	11.8	7162	7098	
	SRR892505	6.8	1.5	11.1	3449	3680	
	SRR892403	7.07	4.7	11.7	3414	3791	
	SRR892407	7.3	4.6	11.1	3328	3419	
		Improvement					
		Lossless Mode	Lossy Mode				
SPRING[32]	Pseudomonas aeruginosa	115	62				High Computational requirements
	Metagenomic	3206	1736				
	H.sapiens	28901	13460				
	H.sapiens	6971	5657				
	H.sapiens	25883	20316				
		CR	CT	DT	Memory Usage		
					C	D	
MZPAQ[33]	SRR554369	7.04	0.98	0.91	2398.8	2383.9	Minimum compression ratio gain
	SRR327342	8.49	1.34	1.07	2901.8	2382	
	MH0001	7.98	1.33	1.29	2691	2384.1	
	SRR1284073	3.22	0.78	0.82	2385.6	2383	
	SRR870667	6.27	0.99	0.97	4544.5	2396.3	
	ERR174310	5.00	0.83	0.99	5326.4	2383	
		CR	CT	DT	MC		
LFastqC[34]	SRR001471	5.29	2m00s	2m16s	4		Does not support color space encoding
	SRR003177	5.15	10m13s	10m43s	4		
	SRR003186	4.71	7m15s	7m59s	4		
	SRR007215	6.60	6m18s	6m08s	4		
	SRR010637	5.30	21m18s	20m59s	4		
	SRR013951	3.46	37m20s	35m27s	4		
	SRR027520_1	4.28	44m37s	48m27s	4		
	SRR027520_2	4.25	46m42s	55m49s	4		

Shubham Chandak et al., (2020) proposed a reference free compression technique for FASTQ files named SPRING. Two different modes are used precisely, lossless mode (default mode) to encode and decode FASTQ files with no loss of information and lossy mode where the arrangement of pairs and read identifiers are discarded. SPRING has achieved better results than other standard algorithms [32].

El Allali and Arshad (2019) developed a special tool called MZPAQ for compressing the genomic data in FASTQ formats. It amalgamates the features of both MFCompress and ZPAQ algorithms. The input sequence is alienated into four streams using MZPAQ. Initially, MFCompress will encode the read identifier and read sequence, next operator plus is removed and finally ZPAQ algorithm is applied. The MZPAQ achieved best compression ratio with high speed and reduced memory requirements [33].

Sultan Al YamiI and Chun-Hsi Huang (2019) proposed a lossless non-reference-based FASTQ compressor (LFastqC) which is an enhanced version of LFQC tool to decrease storage space and transmission time. The tool resulted with an enhanced compression ratio when compared with other standard algorithms. The compressor notably decreased the computation time and obtained an average compression ratio. The major drawback is that LFastqC does not support color space encoding [34].

6. HYBRID ALGORITHMS

This section discusses hybrid algorithms for DNA sequence compression (Table 7). Secure Compression Algorithm for Next Generation Sequencing (SCA-NGS) was described by Muhammad Sardaraz and Muhammad Tahir (2021). General-purpose compression library is utilized to minimize the size of quality score. The method enciphered the compressed data by applying crossover and mutation genetic algorithm concept. Results show that the proposed algorithm achieved better compression ratio of 5.08, 5.48, 5.82, 4.03, 4.65, 5.48, 5.12 and 4.19 bpb when tested with SRR801793 (2818.11), ERR022075 (11253.16), SRR125858 (52172.64), SRR611141 (1799.86), SRR489793 (13132.48), SRR935126 (10039.24), SRR003177 (1672.78) and SRR400039 (65723.77) datasets respectively [35].

Yao et al., (2021) suggested the MtHRCM and HadoopHRCM hybrid referential methods. The MtHRCM method is based on multi thread parallel technology and HadoopHRCM is implemented using distributed computing parallel technology. To assess the performance of the proposed techniques, four genomic standard datasets are chosen namely K131, YH, Huref, and HG00096 from 1000 Genome Project. The proposed methods reduced the file size from 3182 GB to 1322 MB with increased computational speed [36].

Milton Silva et al., (2020) developed a reference free and referential compression called GeCo3. The technique was applied to both multiple context model and substitution-tolerant context model of several order-depths. The algorithm mainly focuses on inputs, updates, outputs, and training process of neural networks. GeCo3 achieved better compression ratio when compared with other standard algorithms but resulted with high computational time [37].

Zeinab Nazemi Absardi and Reza Javidan (2020) proposed an innovative deep neural network based DNA sequence compression algorithm using auto encoder. Initially, the DNA sequence is preprocessed to achieve accurate results. Preprocessing is carried out in three steps. 1) Convert the characters into lowercase. 2) Delete line breaks. 3) Finally, transform non-base characters to character ‘n’. The preprocessed data is now encoded using three bit encoding scheme. A binary array is generated from the binary coded sequences. Using auto-encoder the binary array is trained and compressed. The proposed technique achieved five times better compression ratio with an improved compression accuracy of 92% [38].

Wang et al. (2018) developed DeepDNA which encompasses Convolutional Neural Network (CNN) and Long Short-Term Memory Network (LSTM) to minimize the size of genomic data. Machine learning techniques are implemented to compress the Human mitochondrial genome. The DeepDNA achieved good compression ratio of less than 0.05 bpb when compared with Gzip, MFCompress, and DMcompress [39].

Table 7: Performance Evaluation of Hybrid Compression Methods

Methodology	**Dataset**	**Performance Metrics**					**Drawback**
		CR	**CET**	**CEM**	**DDT**	**DDM**	
SCA-NGS [35]	SRR801793	5.09	180	1148	58	1331	Time taken for encryption is high
	ERR022075	5.48	552	1131	305	1528	
	SRR125858	4.76	2437	1638	1531	2132	
	SRR611141	4.03	102	948	36	1142	
	SRR489793	4.65	876	1536	490	1562	
	SRR935126	5.33	412	1126	193	1433	
	SRR003177	5.12	68	1638	26	1532	
		Compression Size					
MtHRCM/ HadoopHRCM [36]	chr1	108.94					
	chr2	113.25					
	chr3	98.55					
	chr4	90.54					
	chr5	81.99					
	chr6	77.91					
	chr7	73.05					

	chr8	73.56		
	chr9	53.31		
	chr10	59.18		
	chrX	48.86		
	chrY	2.07		

GeCo3[37]		CR	Speed	Execution time is high
	HSxPT	3.65	296	
	HSxPA	6.57	294	
	HSxGG	4.96	293	
	GGxHS	5.81	301	
Deep Neural Network Approach [38]		CR	CT	Training time was high
	KOREF_20090224	4.801	16.692	
	KOREF_20090131	5.104	17.230	
	KOREF_20090224	4.902	27.55	
	KOREF_20090131	5.192	28.215	
	KOREF_20090224	5.003	39.002	
	KOREF_20090131	5.314	39.956	
	KOREF_20090224	5.003	42.318	
	KOREF_20090131	5.318	43.087	
DeepDNA [39]		CR		
	KF162105.1	0.01		
	MF058266.1	0.05		
	KC911416.1	0.01		
	AY339411.1	0.01		
	JQ702777.1	0.04		

7. CONCLUSION

DNA Sequence Compression is a rapidly growing and strongly related field to bioinformatics research frontiers. It is vital to study the key research issues in bioinformatics and develop new algorithms for compressing the DNA sequence for efficient analysis. The paper discusses about the classification of different lossless DNA sequence compression algorithms together with its merits and drawbacks. Some algorithms are not able to reduce the size of DNA sequences (or not achieve good compression ratio). The lossless DNA sequence compression algorithms focused include three different directions, namely, horizontal mode, vertical mode and hybrid. In each direction, different techniques are illustrated along with its experimental results such as compression ratio, time taken for compression and decompression and memory usage. Generally, horizontal mode compression techniques are applied to minimize the size of the sequences. Alternatively vertical mode compression techniques are also used to compress the sequences. Although a broad survey on the taxonomy of various lossless DNA sequence compression algorithms and their effectiveness is well beyond the scope of this survey, the results discussed here may give huge idea to readers that many remarkable works has been carried out in this analysis. Though DNA compression is highly challenging and shows potential direction, remarkable results will appear in future experiments.

8. REFERENCES

[1] Ziv, Jacob, and Lempel, A. (1977), A Universal Algorithm for Sequential Data Compression, IEEE Transactions on Information Theory, Vol. 23(3), pp. 337-343.

[2] Ziv, Jacob, and Lempel, A. (1978), Compression of Individual Sequences via Variable Rate Coding, IEEE Transactions on Information Theory, Vol. 24(5), pp. 530-536.

[3] Cleary, John, G. and Witten, H. (1984), Data Compression Using Adaptive Coding and Partial String Matching, IEEE Transactions on Communications, Vol. 32(4), pp. 396-402.

[4] Willems, F. M. J., Shtarkov, Y. M., and Tjalkens, T. J. (1995), The context tree weighting method: Basic properties, IEEE Transaction Information Theory, Vol. 41(3), pp. 653-664.

[5] Gailly, J. and Adler, M. (1992), gzip (GNU zip) compression utility. [www.gnu.org/software.gzip] website.

[6] Welch Terry, A. (1984), A technique for high performance data compression, IEEE Computer, Vol. 17(6), pp. 8-19.

[7] Julian Seward (1996), bzip2 [sourceware.org/bzip2] website.

[8] Neva Cherniavsky and Richard Ladner (2004), Grammar-based Compression of DNA Sequences, UW CSE Technical Report, pp. 1-21.

[9] Murugan, A. and Punitha, K. (2021), Pattern Matching Compression Algorithm for DNA Sequences, Proceeding of the International Conference on Sustainable Expert System, Vol. 176, pp. 387-402.

[10] Benson, D. A., Karsch-Mizrachi, I. and Lipman, D. J. (2005), GenBank, Nucleic Acids Research, Vol. 33, pp. 34-38.
[11] Murugan, A. and Punitha, K. (2021), A Pattern Matching Extended Compression Algorithm for DNA Sequences, International Journal of Computer Science and Network Security (IJCSNS), Vol. 21(8), pp. 196-202.
[12] Wenwen Cui, Zhaoyang Yu, Zhuangzhuang Liu, Gang Wang, and Xiaoguang Liu (2020), Compressing Genomic Sequences by Using Deep Learning, International Conference on Artificial Neural Networks and Machine Learning (ICANN), pp. 92-104.
[13] Diogo Pratas, Morteza Hosseini, and Armando J. Pinho (2020), GeCo2: An Optimized Tool for Lossless Compression and Analysis of DNA Sequences, International conference on Advances in Intelligent Systems and Computing, Vol. 1005, pp. 137-145.
[14] Hui Chen (2020), Application of Genome Sequence Based on Entropy Coding, International Conference on Intelligent Computing, Automation and Systems (ICICAS), pp. 156-159.
[15] Deloula Mansouri and Xiaohui Yuan (2018), One-Bit DNA Compression Algorithm, Proceedings of the International Conference on Neural Information Processing, Cambodia, pp. 376-386.
[16] Shan E Zahra, Khalid Masood and Muhammad Asif (2019), DNA Compression using an innovative Index based Coding Algorithm, IEEE 978-1-7281-4001-8/19.
[17] Ayad E. Korial and Ali Kamal Taqi (2018), Propose a Substitution Model for DNA Data Compression, International Journal of Computer Applications (0975 – 8887), Vol. 179, pp. 20-26.
[18] Gede Eka Sulistyawan, I., Achmad Arifin and Muhammad Hilman Fatoni (2020), An Adaptive BWT-HMM-based Lossless Compression System for Genomic Data, International Conference on Computer Engineering, Network and Intelligent Multimedia(CENIM 2020), pp. 429-434.
[19] Sebastian Deorowicz (2020), FQSqueezer: k-mer-based compression of sequencing data, Scientific Reports nature research, https://doi.org/10.1038/s41598-020-57452-6.
[20] Sanjeev kumar, Suneeta Agarwal and Ranvijay (2020), WBTC: A new approach for efficient storage of genomic data, International Journal of Information Technology, Springer, International Journal of Information Technology, https://doi.org/10.1007/s41870-020-00472-2.
[21] Vey, G. (2009), Differential Direct Coding: A compression algorithm for nucleotide sequence data. Database, Vol. 2009, Article ID bap013, doi:10.1093/database/bap013.
[22] Raju Bhukya (2019) Modified Direct Differential Coding Using 2D-Dynamic Dictionary for Nucleotide Sequence, Bioscience Biotechnology Research Communications, Vol. 12(4), pp. 1150-1158.
[23] Jothi, S., Chandrasekar, A., and Ranjith, R. (2018), Lossless Segment with Lempel-Ziv-Welch Compression Algorithm Based DNA Compression, Taga Journal, Swansea Printing Technology Ltd., Vol. 14, pp. 1548-1554.
[24] Shengwang Du, Junyi Li, and Naizheng Bian, A compression method for DNA, PLOS ONE, Vol. 15(11), pp. 1-8.
[25] Diego Diaz Dominguez and Gonzalo Navarro, A grammar compressor for collections of reads with applications to the construction of the BWT, IEEE DOI: 10.1109/DCC50243.2021.00016.
[26] Gagie, T., Tomohiro, I., Manzini, G., Navarro, G., Sakamoto, H., and Takabatake, Y. (2019), Rpair: Rescaling RePair with rsync, in Proc. 26th SPIRE, pp. 35-44.
[27] Ferragina, P., and Manzini, G. (2005), Indexing compressed text, J. ACM, Vol. 52(4), pp. 552-581.
[28] Gagie, T., Tomohiro, I., Manzini, G., Navarro, G., Sakamoto, H., Benkner, L., and Takabatake, Y. (2020), Practical random access to SLP-compressed texts, in Proc. 27th SPIRE, pp. 221-231.
[29] Murugesan, G. (2020), Codon Based Compression Algorithm for DNA Sequences with Two Bit Encoding, European Journal of Molecular and Clinical Medicine, ISSN 2515-8260, Vol. 07(10), pp. 33-41.
[30] Bruno Carpentieri (2020), Compression of Next-Generation Sequencing Data and of DNA Digital Files, MDPI, Algorithms, Vol. 13 (151), pp. 1-11.
[31] Anibal Guerra, Jaime Lotero and Jose Edinson Aedo (2020), Tackling the Challenges of FASTQ Referential Compression, Bioinformatics and Biology Insights, Vol. 13, pp. 1-19.
[32] Shubham Chandak, Kedar Tatwawadi, Idoia Ochoa, Mikel Hernaez, and TsachyWeissman (2018), SPRING: A next-generation compressor for FASTQ data, Oxford, Bioinformatics, Vol. 35(15), pp. 2674-2676.
[33] Achraf El Allali and Mariam Arshad (2019), MZPAQ: A FASTQ data compression tool, Source Code for Biology and Medicine, Vol. 14(3), pp. 1-13.
[34] Sultan Al Yami and Chun-Hsi Huang (2019), LFastqC: A lossless non-reference-based FASTQ compressor, PLOS ONE, pp. 1-10.
[35] Muhammad Sardaraz, and Muhammad Tahir (2021), SCA-NGS: Secure Compression algorithm for next generation sequencing data using genetic operators and block sorting, Science Progress, Vol. 104(2), pp. 1-18.

[36] Haichang Yao, Shuai Chen, Shangdong Liu, Kui Li, Yimu Ji, Guangyong Hu, and RuchuanWang (2021), Parallel compression for large collections of genomes, Concurrency Computat. Pract. Exper. John Wiley & Sons, Ltd., pp. 1-13.

[37] Milton Silva, Diogo Pratas, and Armando J. Pinho (2020), Efficient DNA sequence compression with neural networks, Gigascience, Oxford, pp. 1-15.

[38] Zeinab Nazemi Absardi and Reza Javidan (2019), A Fast Reference-Free Genome Compression Using Deep Neural Networks, Proceedings of the 2019 IEEE Conference on Big Data, Knowledge and Control Systems Engineering (BdKCSE), IEEE 978-1-7281-6481-6/19.

[39] Rongjie Wang, Tianyi Zang and Yadong Wang (2019), Human mitochondrial genome compression using machine learning techniques, Human Genomics, Vol. 13(1), pp. 1-8.