A Biological Sequence Compression based on Look up Table (LUT) using Complementary Palindrome of Fixed Size

R.K. Bharti
Research Scholar, UTU, Dehradun
Astt. Prof, CSE Deptt
B.T. Kumaon Institute of Technology, Dwarahat,
Uttarakhand, INDIA-263653

Prof. R.K. Singh
Prof, ECE Deptt
B.T. Kumaon Institute of Technology, Dwarahat,
Uttarakhand, INDIA-263653

ABSTRACT

Data Storage costs have an appreciable proportion of total cost in the creation and analysis of DNA sequences. In particular, the increase in the DNA sequences is highly remarkable with compare to increase in the disk storage capacity. General text compression algorithms do not utilize the specific characteristics of DNA sequences. In this paper we have proposed a compression algorithm based on cross complementary properties of DNA sequences. This technique helps for comparing DNA sequences and also to identify similar subsequences which may lead to the identification of structure as well as similar function.
The experimental results show that it performs better compression as compared to other existing compression algorithms.

Keywords: DNA Sequences, Text Compression, Cross Chromosomal similarity, complementary palindrome, LUT compression.

1. INTRODUCTION

Genomic repositories contain a large amount of data nowadays, due to which need of some efficient algorithms have emerged to facilitate communication and storage. Out of this, cheap storage is not a big issue as due to limited bandwidth communication of modern world. Data compression is a general word with respect to computer science and information technology which particularly means reduction in size of memory used to store data. This reduction can be done by using bit rate reduction in source code, data compression techniques etc.

DNA is an abbreviation for deoxyribonucleic acid which carries hereditary information. Most of the DNA's found in humans are same which is concentrated in cell nucleus, except some which are found in cell's mitochondria. The former one is known as nuclear DNA, while the latter one is mtDNA. There are four different types of nucleotides found in DNA, differing only in the nitrogenous base. They are: A,G,T,C. There are nearly 3 billion bases of human DNA, out of which 99% are same in all humans. Thus, organisms are primarily defined with these bases.

Earlier, a lot of text compression techniques have been applied on biological data, which worked less efficiently. Nowadays huge biological data is available, and in growing in size with passage of time. Hence, the information must be stored and communicated efficiently. Compression algorithms can also define differences in the sequences.

The standard techniques of text compression do not compress these sequences; rather they expand the size of file, whereas the DNA compression techniques compress it for less than 2 bits per DNA base.

BioCompress -1 and BioCompress -2 are compression algorithms uses a window of size of the sequence to detect palindromes and factors of arbitrarily long and far from each other. They encode the factor by the pair (l, p) where l is the length of the factor and p is first occurrence's position. We use two bit encoding, if the size of the code word is greater than the factor. Decompression may demean the algorithm performance, as it requires reference to the starting of the sequence which requires more memory reference.

Another algorithm is GenCompress the performance of which is based on reference sequence selection as approximate matching with edit operations uses this reference sequence for compression. It uses both approximate repeats and reverse complements, and encodes it with length, position and the errors. It does not help in encoding if approximate repeats and approximate reverse generate errors and that is why they used second order arithmetic encoding. Suppose, for input A having two parts B and C let us assume that B has already been compressed and C is not i.e. A=BC. So the algorithm appends some prefix to C (which is an economic method), so as to make it match with some string or a set which exists in B. this is done till C becomes empty.

The next algorithm GenomeCompress (U.Ghoshastide et al, 2005) compresses both repetitive and non repetitive sequences. The algorithm divides the sequence into segments of length four and assigns a five bit binary sequence for four DNA bases and for eight repeated sequence of each bases also. Four used for encoding repetitive sequence and remaining for encoding segment's bases. It is simple and uses less execution time and memory. The techniques mentioned previously give 1.76 bpb (approximately 22% compression). Genome compression proved to be more effective for storage and time complexity. It uses the set of 5 bits which can contain 2^5=32 characters. These 5 bit binary numbers can replace a set of 4 bases. A set of 4 unique 5 bit binary numbers are assigned to AAAAAAAA, CCCCCCCC, GGGGGGGG and TTTTTTTT, and are replaced by these unique numbers whenever encountered.

The next DNACompress(XIN Chen et al, 2002) is a two phase algorithm and uses a tool PatternHunter for finding the repeats. PatternHunter finds complementary palindromes and

approximate repeats with highest score in the first phase and encodes them in second phase. Hence DNACompress involves less searching time. It checks each repeats to see whether it saves bits to encode, if not it will be discarded. At the end all the non-repeats are concatenated together and encoded. The algorithm achieves a compression rate of only 1.72 bits per base. Let there be a finite sequence made up of A,G,T,C which can have many repeats. We only encode those repeats that provide maximum amount of compression.

Searching approximate repeats

Searching approximate repeats is often time consuming, and greedy approach misses long repeats which prohibits from receiving high compression and other techniques. Hence, the PatternHunter is used in this case which searches like blastn, but is faster than it. It can search for all repeats and complemented palindromes also. When PatternHunter finishes up with its job, it is then decided which repeats should be worked upon to ghet the maximum amount of compression.

Encoding repeats

Srinivasa etal proposed an encoding scheme using dynamic programming approach to compress non repetitive DNA parts. The procedure includes two passes. In the first the alphabets A and G are represented by A whereas T and C are represented by T and in the second pass A and C are represented as A whereas G and T are represented by T. Represent the sequence in matrix form. Divide the matrix in such a way that each sub matrix contains exactly one alphabet. Decompression requires the original size of the sequence.

Another compression, which also divides the entirely scanned DNA sequence into factors of length four, is Hashbased (Ateet Mehta et al, 2010)and as its name itself suggests, the algorithm initially builds a hash table and assigns a unique character to each of the factors which act as the hash key. Each factor of length four is assigned corresponding unique characters to each of the factors. But this algorithm doesn't consider any junk characters in the sequence; it doesn't give any priority for repeated sequences and it is not possible to process any part of the sequence without decompressing the entire sequence.

DNA sequence compression algorithm based on fixed length LUT and LZ77(Sheng Bao et al, 2005) works in two phases. In first phase it creates a fixed size look up table which contains a combination of three bits (for best compression) forming 64 combinations, resulting in the fixed size of the table. The symbols which replace these combinations cannot contain A,G,T,C and a,g,t,c as they represent DNA bases.

Later in second phase, these combinations are replaced by the symbols which are assigned in the lookup table hence resulting in encoding using LZ77 algorithm.

Differential Direct Coding (2D) (Gregory Vey,2009) also divides the sequence into factors of length three. It proposes that compression strategies must accommodate large data sets, consist of multiple sequences and auxiliary data. The set of expected symbols for the 2D model are {A, T, G, C, and U}, which removes the burden of explicit declaration of sequence type like DNA or RNA. The representation in terms of triplets makes 2D very tractable to decompression as a polypeptide sequence of amino acids by interpreting the triplets as codons. Even though many algorithms have already been implemented for DNA sequence compression, it's observed that all algorithms achieve compression by considering the sequence as plain English text, which results in junk data. None of them enables part by part sequence decompression.

2. PROPOSED ALGORITHM

An Algorithm consists of two phases. Fist, we shell search for all palindromes in a specific length(3 Base Character). Searching for palindromes done by checking all the possible places in the sequence (in order to be correct and not to miss even one palindrome). The "heart" of the algorithm compare the first letter to the last letter, the second letter to the letter second form the end, etc. (A matches T and C matches G). If we found a palindrome that correlated with our demands, we will print it to the output.

In second phase we apply the LUT base variable length compression algorithm.

Phase 1: A double strand DNA locus whose 5'-to-3' sequence is identical on each DNA strand. The sequence is the same when one strand is read left to right and the other strand is read right to the left. In other words, a region of sequence, that when it's been read left to right it is complementary to the sequence that been read right to left (A match T, and C match G). Approximate Palindrome contain a certain number of mismatches and allow gap. "palindrome fingerprints" -Each DNA sequence has it's unique number, sizes of palindromes, and location in sequence. We find the palindrome sequence as:

Input
1) Sequence, genome of different organisms, text file in a FASTA format .
2) Length of palindrome (one side).
3) Maximum gap between repeated regions.
4) Number of mismatches allowed.
Output
All the palindromes within a specified length range and also a range of mismatch.
The Algorithm :
1.Search for the palindrome within a sub sequence, in the size of MaxSizeequal to three.
2. Each iteration incrementing the size of palindrome, until MaxSize is reached.
3. Shift left of the sequence .

Phase 2:
In second phase we apply the LUT base variable length compression algorithm.

3. RESULT

Table 1: Comparaison between Different Previous Existing Biological Compression Techniques & LZ 77 Compression (Universel) Techniques

Type of Sequences	Original Size(bits) before compression	Size of the sequences after applying various compression algorithm			
		DNA Compress	Gen Compress	Fixed LUT	Univ.(LZ 77)
Gallus β globin	752	272	360	256	568

Goat alanine β globin	**732**	**256**	**352**	**248**	**516**
Human β globin	**752**	**272**	**360**	**256**	**608**
Lemur β globin	**760**	**280**	**376**	**264**	**592**
Mouse β globin	**776**	**280**	**376**	**264**	**608**
Opossum β hemoglobin β- M gene	**760**	**272**	**376**	**264**	**600**
Rabbit β globin	**736**	**264**	**352**	**256**	**560**
Rat β globin	**752**	**272**	**360**	**256**	**600**
Avg	**752.5**	**271**	**364**	**258**	**581.5**

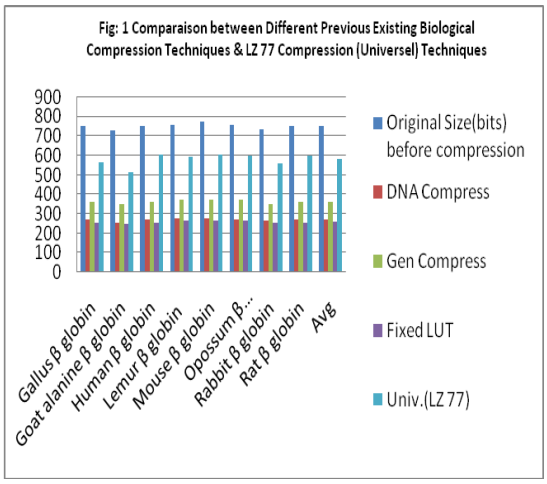


Table 2: Comparison between Different Biological Sequence Compression Techniques with variable LUT

Type of Sequences	Original Size(bits) before compression	Size of the sequences after applying various compression algorithm			
		DNACompress	GenCompress	Fixed LUT	Variable LUT
Gallus β globin	**752**	**272**	**360**	**256**	**248**
Goat alanine β globin	**732**	**256**	**352**	**248**	**232**
Human β globin	**752**	**272**	**360**	**256**	**248**
Lemur β globin	**760**	**280**	**376**	**264**	**256**
Mouse β globin	**776**	**280**	**376**	**264**	**256**
Opossum β hemoglobin β- M gene	**760**	**272**	**376**	**264**	**256**
Rabbit β globin	**736**	**264**	**352**	**256**	**248**
Rat β globin	**752**	**272**	**360**	**256**	**248**
Avg	**752.5**	**271**	**364**	**258**	**249**

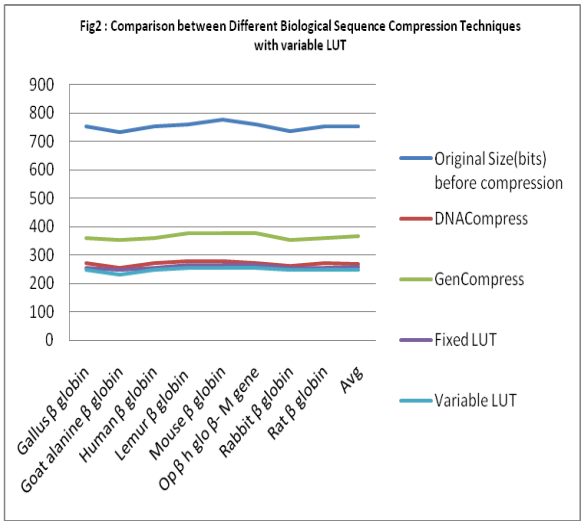
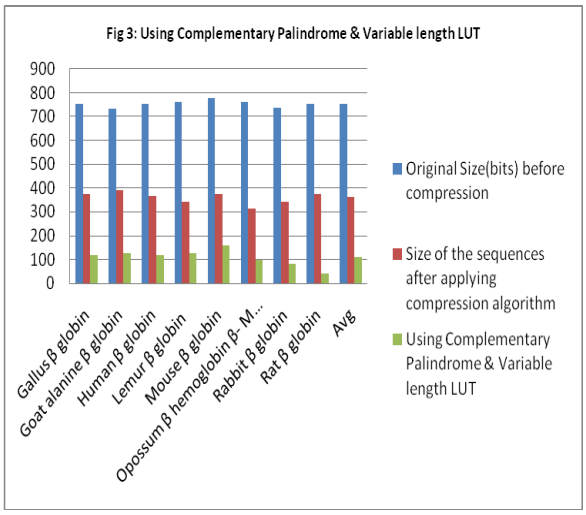


Table 3: Biological Sequence Compression Based on Complementary palindrome Using Variable length LUT

Type of Sequences	Original Size(bits) before compression	Size of the sequences after applying compression algorithm	Using Complementary Palindrome & Variable length LUT
Gallus β globin	**752**	**376**	**120**
Goat alanine β globin	**732**	**392**	**128**
Human β globin	**752**	**368**	**120**
Lemur β globin	**760**	**344**	**128**
Mouse β globin	**776**	**376**	**160**
Opossum β hemoglobin β- M gene	**760**	**312**	**96**
Rabbit β globin	**736**	**344**	**80**
Rat β globin	**752**	**376**	**40**
Avg	**752.5**	**361**	**109**



4. CONCLUSION AND DISCUSSION

There are various universal compression algorithm like LZ77, WinZIP, Win RAR etc but they are not appropriate for the compression of DNA sequences as shown in the experiential results. Therefore specialized DNA sequence compression algorithms were developed like DNA Compress, Gen Compress, 2D, Genome Compress, Fixed length LUT and Variable length LUT. As the result shows that variable LUT gives better result so we choose it for proposed compression algorithm.

Our proposed algorithm has high compression ratio to other exiting Biological Sequence Compression. This algorithm also uses less memory compared to the other existing algorithms and easy to implement.

The proposed algorithm compresses Biological sequences which are complementary in nature. All other algorithm only uses other properties of sequences such as repeated and non repeated. If the sequence is compressed using proposed algorithm it will be easier to make sequence analysis between compressed sequences. It will also be easier to make multi sequence alignment. High compression ratio also suggests a highly repetitive sequence.

5. REFERENCES

[1] Ateet Mehta , 2010, et al., " DNA Compression using Hash Based Data Structure", IJIT&KM, Vol2 No.2, pp. 383-386.

[2] B.A., 2005, " Genetics: A comceptual approach." Freeman, PP 311.

[3] Choi Ping Paula Wu, 2008, et al., " Cross chromosomal similarity for DNA sequence compression", Bioinformatics 2(9): 412-416.

[4] Gregory Vey,2009, "Differential direct coding: a compression algorithm for nucleotide sequence data", Database, doi: 10.1093/database/bap013.

[5] J. Ziv and A.1977, et al., "A universal algorithm for sequential data compression," IEEE Transactions on Information Theory, vol. IT-23.

[6] K.N. Mishra,2010, " An efficient Horizontal and Vertical Method for Online DNA sequence Compression", IJCA(0975-8887), Vol3, PP 39-45.

[7] P. raja Rajeswari, 2010, et al., " GENBIT Compress-Algorithm for repetitive and non repetitive DNA sequences", JTAIT, PP 25-29.

[8] Pavol Hanus, 2010, et al., "Compression of whole Genome Alignments", IEE Transactions of Information Theory, vol.56, No.2Doi: 10.1109/TIT.2009.2037052.

[9] R.K.Bharti,2011, et al., "Biological sequence Compression Based on Cross chromosomal properties Using variable length LUT", CSC Journal, Vol 4 Issue 6, *PP:217-223*.

[10] R.K.Bharti,2011, et al, "Biological sequence Compression Based on properties unique and repeated repeats Using variable length LUT" CiiT journal, Vol 3 Issue, 4, PP: 158 – 162..

[11] R.K.Bharti,2011, et al, "A Biological sequence compression Based on Approximate repeat Using Variable length LUT" International Journal of Advances in Science and Technology, Vol. 3, No.3,PP:71-75.

[12] R. Curnow, 1989, et al. "Statistical analysis of deoxyribonucleic acid sequence data-a review," J Royal Statistical Soc., vol. 152, pp. 199-220.

[13] Sheng Bao, 2005, et al. "A DNA Sequence Compression Algorithm Based on LUT and LZ77".

[14] U. Ghoshdastider,2005, et al., "GenomeCompress: A Novel Algorithm for DNA Compression", ISSN 0973-6824.

[15] Xin Chen, 2002, et al.," DNA Compress: fast and effective DNA sequence Compression" BIOINFORMATICS APPLICATIONS NOTE, Vol. 18 no. 12, Pages 1696–1698.

[16] X. Chen, 2002, et al., "Dnacompress:fast and effective dna sequence compression," Bioinformatics, vol. 18.