DOI: https://doi.org/10.53555/nncse.v3i4.425 **Publication URL: http://nnpub.org/index.php/CSE/article/view/425**

DEVELOPING AN APPROACH TO COMPRESS NON-REPETITIVE CODECS OF DNA USING A NOVEL NDCP: A LOSSLESS UTILITY

Dr. V. Hari prasad[1]*
*[*1]Department of Technical Education, Andhra Pradesh.*
Email: hariprasadvemulapati@gmail.com, drvhp60@gmail.com

Corresponding Author: -
Email ID: - hariprasadvemulapati@gmail.com, drvhp60@gmail.com

Abstract: -

the transformation has been started with Information Theory in the field of Data compression. The outcome of data compression is a technological explosion in internet technology, now the continent is enjoying. Initially the spark of compression has been welded with Text (Lossless) compression and later it has been speeded to the allied areas like Geneti (DNA&mRna) and multimedia data compression (Lossless&Lossy). In 2004 the Human Genome project was deciphered. Can you imagine the human genome requires in an around of 30-35 GB for storage and maintenance? If at all if would have to maintain census data bases the infrastructure would requires substantially larger. So the remedy is compression of genetic sequences. Due to arrival of different DNA sequences the public genetic databases size growing like in an exponential manner. To limit it state of the art many DNA compression algorithms were strived into the compression era, but they work with common performance analysis of best, worst and avg cases based on repetitiveness of the DNA sequences. In case if DNA contains many non-frequent fragments(non-codecs) the existing techniques may run in worst case. A new methodology is highly inevitable for non codecs. In this work a Lossless novel utility Tool NDCP (Non-codecDNAcompression) was proposed to delimit the feasible compression ratios of existing compression techniques.

Keywords: -*compression; encoding; decoding; bio compress; Huffbit compress;dnabit compress;LSBD compression.*

1. INTRODUCTION

In Bio informatics information technology is applied in terms of biological sciences. The greatest achievement in Bio era Human genome project is now completed. In modern molecular biology, the genome can contain hereditary information and encoded its data in terms of non-coding sequences of DNA and genes Due to the excessive surge of genetic sequences in the public databases (such as Genbank or EMBL for primary DNA sequences) their size is increasing in a tremendous growth. So to maintain and distribute such data in client-server processing through web services is a daunting task. Suppose if we can take single person DNA itself cost more than of 100GB, so process this information on different sites on a distributed network consumes more time and space. Hence need of compression becoming a greater challenge for the researchers. Compression comes in two flavors one lossless and other is loss. Loss compression can be applicable for multimedia applications like image, audio and video. In multimedia applications if we remove some unused pixels also resultant may not variant like removing noise from audio or removing unnecessary pixels from images But Text compression is always loss less even after decoding the entire encoded text we have to retain its original property. DNA can be encoded in four letter alphabets like text {A, C, G, and T}.Thus each Base of symbol (Base) can be represented by two bits. General purpose compression algorithms do not perform well with biological sequences. Giancarlo *et al*. [**1**][**2]** have provided a review of compression algorithms designed for biological sequences. Finding the characteristics and comparing Genomes is a major task (Koonin 1999**[3]**; Wooley 1999**[4]**). In mathematical point of view, compression implies understanding and comprehension (Li and Vitanyi 1998) **[5]**. Compression is a great tool for Genome comparison and for studying various properties of Genomes. DNA sequences, which encode life should be compressible. It is well known that DNA sequences in higher eukaryotes contain many tandem repeats, and essentials genes (like rRNAs) have many copies. It is also proved that genes duplicate themselves sometimes for evolutionary purposes. All these facts conclude that DNA sequences should be compressible. The compression of DNA sequences is not an easy task. (Grumbach and Tahi 1994**[6]**, Rivals *et al*. 1995 **[7]**; Chen *et al*. 2000 **[8]**) DNA sequences consists of only four nucleotides bases {a,c,g,t}. Two bits are enough to store each base. The standard compression software's such as "compress", "gzip", "bzip2", "winzip" expanded the DNA genome file more than compressing it.

Many Universal algorithms are failed to compress genetic sequences due to the specificity of "text" because DNA consists similarities in random strings and very few hidden regularities. So the classical algorithms for text compression (Bell *et al*. 1990**[9]**) do not work on DNA sequences and some more substitution techniques compressed the sequences in negative rates. There are many text compression algorithms available having quite a good compression ratio. But they have not been proved well for compressing DNA sequences as the algorithm does not incorporate the characteristics of DNA sequences even though DNA sequences can be represented in simple text form.DNA sequences are comprised of just four different bases labeled A, T, C, and G (for adenine, thymine, cytosine, and guanine respectively). T pairs with A, and G pairs with C. Each base can be represented in computer code by a two character binary digit, two bits in other words, A (00), C (01), G (10), and T (11). At first glance, one might imagine that this is the most efficient way to store DNA sequences. Like the binary alphabet {0, 1} used in computers, the fourletter alphabet of DNA {A, T, C, and G} can encode messages of arbitrary complexity when encoded into long sequences.

The technical blue print of this paper is organized as follows. In section 2, we describe general compression algorithms. In section 3 describes existing algorithms related to the problem domain. In section 4 proposed algorithm with performance analysys. In section 5, comparative study with sample illustration and finally in section 6 conclusion followed by future enhancement.

I. GENERAL COMPRESSION ALGORITHMS

Numerous algorithms are proposed for text compression such as LZ76, LH87, Sto88, ZL77, ZL78.With the spirit of Lempel and Ziv based on the detection of palindromes Grumbach and Tahi[10] developed lossless compression algorithm to compress DNA sequences but to encode and decode it will require more memory references so that performance may degrades.

Gencompress[11] developed by Xin Chen,Sam Kwong and Ming Li based approximate repeat references in DNA sequences but some DNA sequences like rice grass will have very infrequent random repeats .So in such cases the above algorithm not achieved handsome results and the performance is purely based on probability of selection references.

DnaPack[12] which uses hamming distance for the repeats and complementary palindromes and it is implemented by dynamic programming approach. So that it is not simple in design and it will require more time to execute and require more memory requirements also. The algorithm achieves a compression rate in an average of 1.6602.

Genome compress [13] developed by Umesh and Saha for both repetitive and non repetitive DNA sequences. It is a fragment bases compression based on nucleotides available in DNA and assigned by five bit binary number and also assigned five bit binary number for eight repeated sequences of each bases. This algorithm simple in design and take less time for execution but it is applicable when more tandems repeat in DNA.

Some more algorithms are proposed exclusively for non repetitive sequences like Srinivasa at el [17]This algorithm is pair based matrix generation developed in two passes. In two passes every two DNA bases are replaced by single base in i.e. A and G represented by A and G and T represented by T and by doing it in the reverse process they achieved original sequence in decoding. Due to Dynamic programming its implementation is complex and requires more memory references.

The Lossless segment based compression enables part by part decompression by introducing non base character so that it will save memory requirements but it is applicable well on repeating sequences are more and more in the sequence. If such sequences like AT-rich DNA, which constitutes a distinct fraction of the cellular DNA of the archaebacterium Methanococcus voltae, consists of non-repetitive sequences, so part by part decompression is little bit tedious.

DNABITcompress [13] will work on approximation of repeats if number of tandem repeats more it saves bits to encode if not discard. Non repeated sequences will be appended to the sequence at the end. This algorithm achieves a compression rate only 1.72 bits per base. If there is no tandem repeat in the sequence it may run in worst case.

III. RELATED EXISTING ALGORITHMS

DNA compression is always is always lossless we have to retain its original property after decoding. In this connection two sorts of universal text compression algorithms are came into existence one is statistical and other is substitution .First one, long code word are replaced by a shorter one for frequent patterns like Huffman coding but this technique is applicable for strings with limited length, an adaptive Huffman also proposed by its not compressed the text much. Second, Dictionary based replacing larger strings by shorter code like Cfact which searches the longest exact matching repeat using tree data structures.

With the spirit of substitution and statistical techniques many lossless Genome compression algorithms are strived based on two bits encoding scheme i.e.[00],C[01],G[10] and T[11]. Some of the algorithms like HUFFBIT [14], GENBIT [15] and DNASC [16] compress given performance analysis (Best, Avg and Worst) cases based on the tandem repeats in the sequence. Obviously if more fragments are repeated we can achieve best case if not average case and worst case. The above algorithms are explained time complexities in terms Big O ,Omega and Theta notations .According to the sequence compression rate is also increases , comes under Best case i.e. O(n) (where n is the length of the sequence) if not algorithms may run in O(nlogn) i.e worst case means less tandem repeats in the sequence and length of the sequence is increased compression rate decreases. The algorithms of [15][16] achieves the compression ratios 1.727 in avg case and 2.323 in worst case and these algorithm most suitable for repetitive DNA sequences in genomes. Through DNASC achieved the compression rate 1.61 in best case and compression rate increases with sequence if the sequence contain tandems repeat.

IV. PROPOSED ALGORITHM

The proposed Utility will work in two stages. In first stage the Input sequence converted into binary Huffman Tree and the output of first stage codec will be the Input for the second stage of NDCP.

A. Idea behind the algorithm

Every DNA sequence contain {A, C, G, T} nucleotides where each literal is named as BASE and encoded in two bits as follows

Compression ratio is calculated encoded bits per Bases.

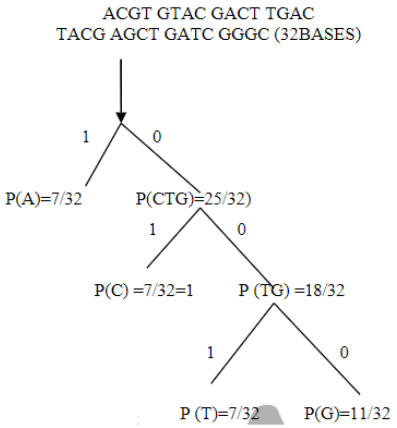
Compression Ratio = Encoded Bits/Bases.

B. Plan of work

First stage: -

Let us consider a sample sequence and construct the Huffman tree for minized codecs.Here the sequence is divided into equal fragments(each fragement contains four bases)

Sequence: -



Now the entropy of Huffman codec is noted as follows

P(A)=1 P(T)=001

P C)=01 P(G)=000

So the above sequence can be encoded in 84 bits and compression raio is nearer to 2.5bpb still it is compressible. The above sequence contains non repetitive fragments so the existing compression algorithms like Huffbit[14] can compress in worst case i.e 2.5bpb.

In our proposed NDCP methodology the first stage o/p will be the input for the second stage for better compression Ratio.

Second stage:

Our algorithm is array based implementation dynamic technique and the compression rate vary with the length of the sequence. NDCP method will work as follows. Re substitute the Huffman Codec bits in the above sequence and prepare a stream of bit pattern as input for the second stage.

$N = length\ of\ the\ bit\ stream\ (84bits)$
$Ncfb = Noncoded\ first\ partitioned\ block\ (16bits)$
$Ncsb = Noncoded\ second\ partitioned\ block\ (16bits)$
$FAb = FirstAvg\ block$
$Cab = Cummulative\ avg\ block$
$Cabv = Cummulative\ avg\ block\ value$
$Esb = Encoded\ segmented\ bits$
$Cr = Compression\ ratio$

Ncfb and Ncsb is calculated as follows .The numeric equivalent value of binary substitution.

$$Ncfb_v = \sum_{b=0}^{p} (Ncf_b)$$

$$Ncsb_v = \sum_{b=0}^{p} (Ncs_b)$$

Here the avg of first and second can be calculated as follows.

$$Fab = \sum_{b=0}^{p} (Ncfbv + Ncsbv / 2)$$

Now the cumulative value for Cab and Cabv calculated as follows.

Cab=Fab1+Fab2+……..+ n

$$Cab_v = \sum_{b=0}^{n} (Cab)$$

Here n is the length of the given sequence and total number of encoded bits to represent the genome sequence is equivalent to E_{sb}

Compression ratio can be computed as follows

$$Cr = E_{sb} / N$$

C. Analysys

Cabv=Fab1+Fab2+Fab3
Esb=2+2+2=6bytes=48bits

Finally, we can calculate the compression ratio in terms of bits per bases.

Cr= 48/32=1.5bpb (bitsperbase)

Compression and Decompression algorithms for DNA sequence is as follows.

D. Encoding Algorithm

INP: input String
OPS: Encoded String
PROCEDURE ENCODE Begin

- Group INS into equivalent fragments as four bases
- Generate all possible combinations of DNA and it will contain non- repetitive and repetitive Every Ncfb and Ncsb contains n/16 fragment bases
- Assign binary bits of Huffman codec of first phase
- Calculate Ncfbv for every Ncfb in INP till eof INP. Ncfbv represents binary equivalent numeric of Ncfb.
- Calculate Ncsbv for every Ncsb in INP till eof INP. Ncsbv represents binary equivalent numeric of Ncsb.
- Calculate Cab for every Avb till eof of INP
- Calculate E_{sb} for every S_b till eof INP

- Repeat the steps 4 and 5 until the length of the INP
- Transfer the sequence Esb to the output string i.e. OPS String. End.

E. Decoding Algorithm
INP: input String
OPS: Decoded String

PROCEDURE DECODE
Begin
- Generate all possible combinations of (A,C,G,T)
- Read the binary data of each sub partition from OPS and assign the two bits by equivalent Base s of Huffman codec) and then store it in an array till eof
- Repeat step 2 until eof INS is reached and calculate Dsb and Db in the reverse process..
- Transfer the sequence D_{sb} to the input String i.e. IN , End

V. EXAMPLE AND COMPARISON

Let us consider the sequence.
Sequence1:
ACGT GTAC GACT TGAC
TACG AGCT GATC GGGC (32BASES)
Sequence length (no of bases) = 32.
Bytes required to store in a text file = 32 Bytes.
Bytes required in ASCII representation=28bytes

The above sequence doesn't contain tandem repeats so existing algorithms like Huff bit compress, Genbit Compress and Dnabit compress may run on worst case and require more bits to encode the sequence.

Huffbit, GenBit and Dna compress =90 bits (2.428)
Genbit Compress (Tool based) = 94 bits (2.404)
NDCP TOOL BASED =48 Bits (1.5bpb)

We compared of our technique with existing techniques and found to be the first-one technique .Proposed technique can be applicable for and non repetitive DNA.

VI. CONCLUSION AND FUTURE WORK

By using of our algorithm we can encode every base by 1.498bits .By applying of ours we are saving nearer of 6 bytes to encode the given sequence, compression may vary with size of the sequence. In addition to that existing techniques uses dynamic programming to compress the sequence which is complex in implementation and time consuming. Our technique is implemented without dynamic programming approach, so it is simple and fast. The simplicity of this will reduce the complexity in processing and definitely it will be the invaluable tool in Bio informatics era. Our algorithm can be extended to any tool based approach.

ACKNOWLEDGMENT

I would like thank my better half G.Lakshmi Prasanna who bared me with utmost compassion and patience by giving 360º support and my replica new born baby boy given me additional boost in extending variant ideas in Bio Informatics era.

REFERENCES

[1].E Schrodinger. *Cambridge University Press*: Cambridge, UK, 1944.[PMID: 15985324]
[2].R Giancarlo *et al. A synopsis Bioinformatics* 25:1575 (2009) [PMID:19251772]
[3].EV Koonin. Bioinformatics 15: 265 (1999)
[4].JC Wooley. J.Comput.Biol 6: 459 (1999) [PMID: 10582579]
[5].CH Bennett et al. IEEE Trans.Inform.Theory 44: 4 (1998)
[6].S Grumbach & F Tahi. Journal of Information Processing and Management 30(6): 875 (1994)
[7].E Rivals et al. A guaranteed compression scheme for repetitive DNA sequences. LIFL, Lille I University, technical report IT-285 (1995)
[8].X Chen *et al.* A compression algorithm for DNA sequences and its applications in Genome comparison. In Proceedings of the Fourth Annual International Conference on Computational Molecular Biology, Tokyo, Japan, April 8-11, 2000. [PMID: 11072342]
[9].TC Bell *et al.* Newyork:Prentice Hall (1990)
[10]. J Ziv & A Lempel. *IEEE Trans. Inf. Theory* 23: 337 (1977)
[11]. A Grumbach & F Tahi. In Proceedings of the IEEE Data [12]
[12]. DNA compression is challenge is revisited Beshad Behajadi
[13]. Allam AppaRao.In proceedings of the Bio medical Informatics Journal [2011].DNABIT compress-compression of DNA sequences
[14]. Allam AppaRao.In proceedings of the *JATIT* journal computationalf Biology and Bio Informatics:[2009].HuffBit compress-compression of DNA using extended binary trees

[15]. Allam AppaRao.In proceedings of the *JATIT* journal computational Biology and Bio Informatics:[2011].Genbit compress-compression of DNA sequences.
[16]. Edries Abdelhadi In proceedings of the *IJCA* journal of computer applications[2010]: An efficient horizontal and vertical method for online *DNA* sequance compression
[17]. Srinivasa K G,Jagadish M, Venugopal K R and L M Patnaik "Efficient compression of non repetitive DNA sequances using Dynamic programming " pages 569-574 IEEE 2006.

Dr.V Hari Prasad ,
B.Tech CSE from JNTU University,Anantapur,M.Tech CSE from JNTUCEH,HYD and welded his PhD in CSE from JNTU KAKINADA, A.P .He has 13 years of teaching experience in various Engineering colleges. Presently He is working in Department of Technical Education A.P. He is a Life Member of MISTE and Member of IEEE. He presented papers at International & National conferences on various domains. His interested areas are Bio Informatics, Databases, and Artificial Intelligence