A Novel DNAZIP Tool for Zipping of Genome Sequences by Linear Bounded Data Structure

Hari Prasad .V
Research Scholar in CSE
Jawaharlal Nehru Technological University
Kakinada, Andhra Pradesh

P.V.Kumar
Department of CSE
Osmania University
Hyderabad, Andhra Pradesh

ABSTRACT

In Of late due to excessive accumulation of genetic sequences need of vacuuming plays predominant role in database when it reaches to its threshold. Vacuuming refers to data should not be deleted physically but it is superseded. The achieved data can be stored in secondary storage, we can access it by writing temporal quarries but applying temporal quarries to genome data is spiky process. Hence need of zipping arises for processing, storing and managing data in a centralized and distributed environment. The genome is living organism encoded in DNA and RNA .Many existing algorithms had a eye on genome compression but they are having their own tradeoff in their compression ratios.DNA may be repetitive and non repetitive in nature based on this property earlier algorithms performed and achieved bountiful results. In this paper, we describing a novel DNASCP tool based on linear bounded array data structure for easy preservation of lossless property and secured transmission of data in a client server environment..Our tool will work on both repetitive and non repetitive sequences and computed results are on par with existing techniques.Definitely our one will becomes an invaluable tool in bio informatics era.

Keywords

Compression, Encode, Decode, push down stack, Huff bit, GenBit,, DNASC,DNABIT,LSBD compression.

1. INTRODUCTION

Bio Informatics is playing a predominant role in DNA forensics in identification of personal identity of criminal investigation. In addition to that the field bio informatics is scattered to molecular medicines, rational drug design, bio archeology, anthropology and human migration. Bio informatics is nothing but information technology is applied in terms of biological sciences. The greatest achievement of bio informatics is human genome project is now deciphered with 23 chromosome pair over 3 billion pairs of characters with more than 32 GB of data surged in public database NCBI. 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. Throughput is effecting in centralized and distributed system for processing such information locally or globally at available sites. Hence need of compression arises for compaction of genomes so computational biology is strived into the vogue. 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 four-letter alphabet of DNA {A, T, C, and G} can encode messages of arbitrary complexity when encoded into long sequences.

1.1 Plan of the Paper

This paper is organized as follows. Section 2 describes basic knowledge of DNA and RNA section 3 describes general compression algorithms. Section 4 describes related existing algorithms to compress genome data. Section 5 describes proposed algorithms analysis how it is better one than existing techniques. Section 6 describes comparative study on a sample sequence. Section 7 is concluding with future work.

2. BASIC KNOWLEDGE OF GENOME DATA (DNA AND RNA)

The complete set of genetic information for a cell is referred to as its **genome.** Technically, this includes plasmids as well as the chromosome; however, the term genome is often used interchangeably with chromosome. The genome of all cells is composed of DNA, but some viruses have an RNA genome. The functional unit of the genome is a **gene.** A gene encodes a product, the **gene product,** most commonly a protein. The study of the function and transfer of genes is called **genetics,**

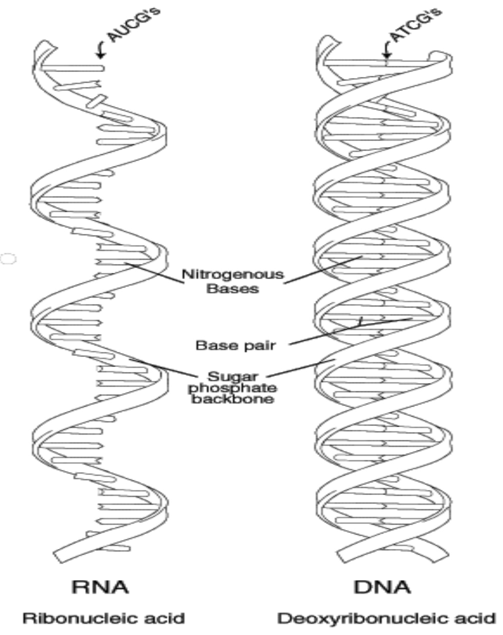

Fig.1: DNA & RNA Helical structure

A single strand of DNA is composed of a series of deoxyribonucleotide subunits, more commonly called nucleotides adenine (A), thymine (T), guanine (G), or cytosine (C). The DNA in a cell usually occurs as a double-stranded, helical structure**.** The two strands are held together by weak hydrogen bonds between the nitrogenous bases of the opposing strands. While individual hydrogen bonds are readily broken, the duplex structure of double-stranded DNA is quite stable because of the sheer number of bonds that occurs along its length. Because short fragments of DNA have correspondingly fewer hydrogen bonds, they are readily separated into single-stranded pieces. Separating the two strands is called **denaturing.**

The two strands of double-stranded DNA are complementary**.** Wherever an adenine is in one strand, a thymine is in the other; these two opposing nucleotides are held together by two hydrogen bonds between them. Similarly, wherever a cytosine is in one strand, a guanine is in the other. These are held together by the formation of three hydrogen bonds, a slightly stronger attraction than that of an A:T pair. The characteristic bonding of A to T and G to C is called **basepairing** and is fundamental to the remarkable functionality of DNA. Because of the rules of base-pairing, one strand can always be used as a **template** for the synthesis of the complementary opposing strand.

While the two strands of DNA in the double helix are complementary, they are also **antiparallel.** That is, they are oriented in opposite directions. One strand is oriented in the 5,, to3,, direction and its complement is oriented in the 3,, to 5,, direction. This also has important implications in the function and synthesis of nucleic acids.

RNA is in many ways comparable to DNA, but with some important exceptions. One difference is that RNA is made up of ribonucleotides rather than deoxynucleotides, although in both cases these are usually referred to simply as nucleotides. Another distinction is that RNA contains the nitrogenous base uracil in place of the thymine found in DNA.

Like DNA, RNA consists of a sequence of nucleotides, but RNA usually exists as a single-stranded linear molecule that is much shorter than DNA. A fragment of RNA, a **transcript,** is synthesized using a region of one of the two strands of DNA as a template. In making the RNA transcript, the same base-pairing rules of DNA apply except uracil, rather than thymine, base-pairs with adenine. This base-pairing is only transient, however, and the molecule quickly leaves the DNA template. Numerous different RNA transcripts can be generated from a single chromosome using specific regions as templates. Either strand may serve as the template. In a region the size of a single gene, however, only one of the two strands is generally transcribed. As a result, two complementary strands of RNA are not normally generated.

DNA can be converted to RNA simply replacing thymine T by uracil U in ribonucleic acid. In the below figure (ii) shows how the sample sequence of DNA converted to mRNA.

DNA
ACGT GCGC GATC GCCT GCTA GGCG TACG TCGC AGGC GATC GATG TGCT AGAT CAGA TGAC TCAG TGCA CGAT.

mRNA
ACGU GCGC GAUC GCCU GCUA GGCG UACG UCGC AGGC GAUC GAUG UGCU AGAU CAGA UGAC UCAG UGCA CGAU.

Fig.(ii)

The conversion process is much useful in central dogma of molecular biology i.e DNA to RNA and RNA to PROTEIN in natural evaluation processes of transcription and translation process which is useful in DNA replication.

3. GENERAL COMPRESSION ALGORITHMS

The compression of DNA sequences is considered as one of the most challenging tasks in the field of data compression. In this connection the very first DNA compression and its subsequent algorithms BioCompress[10] and BioCompress-2[11] detects exact repeats and complementary palindromes located in the sequence and Encode the factor by the size representation(l, p) where l is the length of the factor and p is the position of its first occurrence .If the size is greater the factor then use two bit encoding . More memory references will require decoding the same, so the performance may degrade. In addition to that some of the lossless algorithms CTW, Gen2, and GeNML are also available to compress DNA sequences but they are never achieved higher efficiency for longer sequences.

Gencompress [12] is a lossless compression algorithm for genetic sequences based on searching for approximate repeats of hidden regularities of DNA sequences .This algorithm achieved compression rate of 1.800 bits per bases in an average.

DnaPack[13] 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 1.777.

DNASC [14] was developed by applying both substitution and statistical methods based on 128 conditions to preserve loss property of genome sequences. In this technique they showed how transformation can be performed from one location to another location. This technique is inspired by Gen2 and GeNML of horizontal and vertical methods of compression data [Karodi and Tahi] and achieved compression rate of 1.501 in an average for both repetitive and non repetitive sequences.

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 LSBD [16] 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.

3.1 Related Existing Algorithms

Dna compression is always loss less ,we have to retain ite original property after decoding. Zipping and unzipping is one of the secured mechanism we can use the DNA data transmission from one location to another location. Most of the existing techniques mainly classify into one is substitution and other is statistical. First mechanism is replacement of short code by the longer sequences and other is dictionary based mechanism. With the spirit of substitution and statistical techniques many lossless Genome compression algorithms are strived based on two bits encoding scheme i.eA.[00],C[01],G[10] and T[11]. Some of the algorithms like DNASC [14] and DNABITcompress [15] 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 1.583 bits per base.

Our proposed new algorithm is developed based on comparative study of existing techniques and achieves better compression rate i.e. 1.398 which is on par of existing ones. Our algorithm is applied on both repetitive and non repetitive sequences of DNA and achieved the same compression rate in best, avg and worst cases. Really it is a first-rate technique it is achieving the same compression in all the cases and it is very simple in design and it will take less time for execution also. In implementation our technique performance is varying in a linear way in all the cases proportional to the length of the sequence even the sequence may contain tandems repeat of bases in DNA (DNA composed of text bases like alphabet and every alphabet is called as base) .

3.2 Comparison Results of Existing Algorithms

General compression algorithms are computed their compression ratios in terms of bits per bases. Existing algorithms work on two mitochondria: MPOMTCG, PANMTPACGA (also called MIPACGA); two chloroplasts:

CHNTXX and CHMPXX (also called MPOCPCG); sequences from humans: HUMGHCSA, HUMHBB, HUMHDABCD, HUMDYSTROP, HUMHPRTB,complete genome from the two viruses: VACCG and HEHCMVCG .Compression results is as shown in fig(iii).

Table 1. Existed compression algorithms

Sequence	**size**	**CTW**	**Bio2**	**Gen2**	**Dna Pack**	**Dnasc**	**Dnabit**
CHMPXX	121024	1.838	1.684	1.673	1.660	1.500	1.517
CHNTXX	155844	1.933	1.617	1.614	1.610	1.510	1.584
HEHCMVCG	229354	1.958	1.848	1.847	1.834	1.800	1.573
HUMDYSTROP	33770	1.920	1.926	1.922	1.908	1.890	1.572
HUMHBB	73308	1.892	1.88	1.820	1.777	0.910	1.606
HUMHDABCD	58864	1.897	1.877	1.819	1.739	1.610	1.606
HUMHPRTB	56737	1.913	1.906	1.846	1.788	1.710	1.574
MPOMTCG	186609	1.962	1.937	1.905	1.893	1.880	1.565
VACCG	191737	1.857	1.761	1.761	1.758	1.700	1.652
AVG		1.907	1.796	1.800	1.774	1.612	1.583

Fig. (iii)

4. PROPOSED ALGORITHM

Our proposed algorithm is developed based on comparative study of existing techniques. Our technique DNAZIP (DNA zipping) is implemented by using linear data structure stack array. In our technique we are using bi stage stack array for encoding and decoding process to preserve lossless property. We are taking DNA in FASTA format as input for our algorithm. Our technique can be compared with existing and found the first-art results 1.358 bits per bases in an average which is on par of existing techniques. This technique can be applied on both repetitive and non repetitive sequences of DNA and observed the same compression rate on it.

4.1 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

A=00, C=01, G=10 and T=11.

Compression ratio is calculated encoded bits per Bases.

Compression Ratio = Encoded Bits/Bases

4.2 Plan of work

Our algorithm work in two stages, one is zipping and other is unzipping. Here we are taking a sample DNA sequence in FASTA format and divided into n/4 fragment bases (where each fragment contain four bases).In the first stage we can group triplet fragments into subtrahend and minuend i.e. S_{th} and M_{th} ,these two can stored in bi stage array indexes. So Sth and Mth can contain altogether n/24 fragment bases. We can consider these n/24 fragment bases is first segmented values and sub sequent segmented values can calculated by grouping all segments. Clustering of segmented values together in Teb i.e. total number of encoded bits for an entire sequence. We can calculate both Stv and Mtv based on the linear data structure stack array by substituting n/24 fragement bases equivalent binary bits.

Suppose if we took the sample sequence of DNA contain 72 bases ,in the first stage is divided into 18 fragements,6 triplet fragments,3 subtrahend headers , three minuend headers,3 segments and finally grouped to main segment which will represent total encoded bits. Our DNAZIP will work as follows.

$$n = Length\ of\ the\ given\ sequnce$$

$$Sth = Sub\ trahend\ Header$$

$$Sth = Minuend\ Header$$

$$Stv = Sub\ trahend\ Header\ value$$

$$Stv = Minuend\ Header\ value$$

$$Ps = Partition\ set\ contain\ segment\ values$$

$$Mp = Main\ partition\ value$$

$$Teb = Total\ number\ of\ encoded\ bits$$

$$Cr = Compression\ Ratio$$

S_{th} and M_{th} can calculate as follows.

$$S_{th} = S_{t1} + S_{t2} + S_{t3} + ... + S_{tn}$$

$$M_{th} = M_{t1} + M_{t2} + M_{t3} + ... + M_{tn}$$

Here every subtrahend and minuend header contains triplet equivalent binary numeric and we can store both S_{th} and M_{th} indexes in two separate arrays. Now we can calculate Stv and M_{tv} and then grouped into Partition set P_s which will contain set of partitions. Where every partition value contains Stv and U_{tv} numeric equivalent binary value i.e n/24 fragment bases.Ps value also stored in resultant array index.

$$S_{tv} = \sum_{h=0}^{n/12} S_{th}$$

$$M_{tv} = \sum_{h=0}^{n/12} M_{th}$$

Partition set Ps calculate as follows

$$P_S = P_1 + P_2 + P_3 + ... + P_n$$

$$P_1 = (S_{tv} + M_{tv}) / 2$$

$$M_p = \sum_{s=0}^{m} P_s$$

Here Mp will represent the binary equivalent numeric (nearest to integer) in terms of Bytes storage. (Suppose if we will implement the technique in C language unsigned long will require 4 bytes of storage).Here m=n/24 i.e. length of the given sequence.

Total Number of Encoded group bits are calculated as follows.

$$T_{eb} = \sum_{p=0}^{n} (M_p)$$

Finally compression Ratio calculated as follows

$$Cr = (Teb / N)$$

4.3 Analysis

Let us take the sample sequence of DNA will contain 72 bases.

ACGT GCGC GATC GCCT GCTA GGCG TACG TCGC AGGC GATC GATG TGCT AGAT CAGA TGAC TCAG TGCA CGAT.

The above sequence is non repetitive which is scattered into 18 non repetitive fragments,3 subtrahend triplet fragments and 3 minuend triplet fragments grouped into 3 partitions(where each partition is n/24 fragments) which is joined to one main partition set as per earlier explanation.

$$P_s = p_1 + P_2 + P_3$$

We will calculate binary equivalent numeric (which is nearer to integer) value for each partition in terms of Bytes storage (suppose if we will store each partition in C we will require two bytes if we can accommodate on int if not we can go for unsigned long).

Now the sample data is extended to DNAZIP algorithm, the following sequence of digits as follows.To calculate the main partition value we have to calculate three partition values based on S_{tv} and M_{tv} values. Now the first partition contains n/24 fragments and the remaining is n/48.Now S_{TV} and M_{tv} calculated as follows.

000110111001100110001101
110001101101100100101001
001000111000100011100001

100101111001110010011001
100111011001111011100111
110100101110010001100011

Now we can calculate the binary equivalent numeric and we can store the same in one stage stack index. Partition set can be calculated by above formulas.

$$P_s = \begin{pmatrix} 113037 \\ 9936025 \\ 13031721 \\ 9277159 \\ 2328801 \\ 13821027 \end{pmatrix} \quad M_p = \begin{pmatrix} 5024531 \\ 11154440 \\ 8074914 \end{pmatrix}$$

Every partition may contain 24 bases so it may not fit in integer so that we can store in unsigned long. So totally our sequence is divided into three partitions and grouped as one set .So totally we require 12 bytes to store.

$$T_{eb} = P_s(P_1 + P_2 + P_3)$$
$$Teb = (4 + 4 + 4 = 12)Bytes(96bits)$$

Finally we calculate Compression Ratio as follows.

$$Cr = Egb/N$$

$= 96 / 72$
$= 1.333$ (bits per Bases.)

Encoding and decoding algorithms for DNA compression is as follows.

4.4 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 (our INS assumed as no tandem repeats).
- Group six fragments into partition set which will consist of two subpartions.
- Assign binary bits(0&1) for every base of DNA like A=00, C=01, G=10 and T=11
- Calculate Gs for every Ps in INP till eof INP
- Calculate Egb for every Gs till eof INP
- Repeat the steps 4 and 5 until the length of the INP
- Transfer the sequence Egb to the output string i.e. OPS String.

End.

4.5 Decoding Algorithm

```
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 Bases (00=A,01=C,10=G and 11=T) and then store it in an array till eof
  • Repeat step 2 until eof INS is reached and calculate Dgb and Ds in the reverse process..
  • Transfer the sequence Db to the input String i.e. INP
  End.
```

sequence. If more repeats are there in the sequence i.e. best. Case, frequent repeats avg case and if the sequence is non repetitive it may run in worst case. But such sequences like AT-rich DNA, which constitutes a distinct fraction of the cellular DNA of the archaebacterium Methanococcus voltae, and grass plants rice consists of non-repetitive sequences, so earlier compression techniques are not achieved handsome compression rates due to non repetitive fragments are more and more in the sequence and existed algorithms may run in the worst case comparison.

Table 2. DNAZIP comparison of Results with DNASC, DNABIT and other algorithms

Sequence	**size**	**CTW**	**Bio2**	**Gen2**	**DnaPack**	**Dnasc**	**Dnabit**	**DNAZIP**
CHMPXX	121024	1.838	1.684	1.673	1.660	1.500	1.517	1.330
CHNTXX	155844	1.933	1.617	1.614	1.610	1.510	1.584	1.333
HEHCMVCG	229354	1.958	1.848	1.847	1.834	1.800	1.573	1.345
HUMD-YSTROP	33770	1.920	1.926	1.922	1.908	1.890	1.572	1.334
HUMHBB	73308	1.892	1.88	1.820	1.777	0.910	1.606	1.383
HUMHDABCD	58864	1.897	1.877	1.819	1.739	1.610	1.606	1.338
HUMHPRTB	56737	1.913	1.906	1.846	1.788	1.710	1.574	1.485
MPOMTCG	186609	1.962	1.937	1.905	1.893	1.880	1.565	1.333
VACCG	191737	1.857	1.761	1.761	1.758	1.700	1.652	1.357
AVG		1.907	1.796	1.800	1.774	1.612	1.583	1.359

Fig. (iv)

5. EXAMPLE AND COMPARISION

By applying DNAZIP tool o n a sample sequence of DNA (non repetitive) we got 1.33 bits per bases. It means that every base can encoded by 1.33 bits which is on par of existing techniques. The performance of the algorithm varies linearly with the sequence. Our technique can be applicable on both repetitive and non repetitive sequences DNA and it is achieving the same compression rate i.e O (n) in all the cases. This tool applied on different viruses, chloroplasts, human genome sequences and complete genomes of mitochondria [18] and performance results as shown in fig (iv).Finally the conclusion results of DNAZIP, the compression results of *for* DNA sequences indicate that our method based on simple linear data structure implementation without using dynamic programming.DNAZIP is one of the first art technique and achieve best compression results by using this observation.

All most all the existing algorithms based on complementary palindromes, approximate repeats, regularities and grammar based compression. Many statistical and substitution methods proposed to compress DNA sequences. In recently some of the algorithms like HUFFBIT [19] and GENBIT [20] given performance analysis based on tandem repeats in the

Our DNAZIP algorithm also one type of statistical dictionary based approach. Our algorithm is very simple to implement and it is suitable for both intra and internetworking client server environment. Through encryption and decryption we are going to regain the original sequence. Encrypted dna file is very less in size so it's easy navigate in an around networking services, so the throughput of the system will increase. Definitely our one is in valuable tool in bio informatics.

6. CONCLUSION AND FUTURE WORK

By using of our algorithm we can encode every base by 1.33 bits .By applying of ours we are saving nearer of 8 bytes to encode the given sequence, compression may vary with size of the sequence. So our technique is far better than existing ones and we can apply this technique on non repetitive DNA sequences of genomes .If the given sequence can contain tandem repeats also our technique will achieve same compression rate in an average. 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. Our algorithm can be extended to any tool based approach.

7. ACKNOWLEDGEMENT

We would like thank other members of the Bio –Informatics teams (Faculty of CSE and IT) at Sphoorthy Engineering college Nadergul(V),R.R Dist,Hyderabad. We wish to thank Prof&Dr.C.Raghavendra Rao HCU for his esteemed guidance during development phase. We glad to thank of our Chairman & Secretary Sri S.Chalama Reddy & S.Jagan Mohan Reddy extended their valuable support and last but not least students of CSE&IT in sphoorthy for sharing their ideas with us in refining of our architecture.

8. 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] X Chen *et al.* In Proceedings of the Fourth Annual International Conference on Computational Molecular Biology, Tokyo, Japan, April 8-11, 2000.

[13] X Chen *et al. Bioinformatics* **18**: 1696 (2002) [PMID: 12490460]

[14] An Efficient Horizontal and Vertical Method for Online DNA Sequence Compression in IJCA proceedings 2010 vol.3,Issue 1 June 2010.

[15] Allam AppaRao.In proceedings of the Bio medical Informatics Journal [2011].DNABIT compress-compression of DNA sequences

[16] Loss less segment based compression in IEEE confernece proceedings in ICECT-2011 kanyakumari,India.

[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

[18] National Center for Biotechnology Information, Entrez Nucleotide Query, http://www.ncbi.nlm.nih.gov/htbin-post/Entrez/query?db=n s.

[19] Allam AppaRao in proceedings of the JATIT journal computational biology and Bio informatics:[2011].Huffbit compression of DNA sequances

[20] Allam AppaRao in proceedings of the JATIT journal of computational Biology[2011],Genbit compress fro DNA sequances.