Volume 3, Issue 10, October 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

State of the art: DNA Compression Algorithms

Mr Deepak Harbola[1],
Research Scholar,
UTU, Dehradun Programmer,
Computer science & Engineering,
B.T. Kumaon Institute of Technology,
Dwarahat, Almora, Uttarakhand, India

Dr. R.K. Bharti[2]
Assistant Professsor
Computer science & Engineering,
B.T. Kumaon Institute of Technology,
Dwarahat, Almora, Uttarakhand,
India

Abstract: Modern biological science produces vast amounts of genomic progression data. This is igniting the need for competent algorithms for sequence compression and investigation. Data solidity and the related techniques coming from information theory are often perceived as being of great interest for information communication and storage. In recent years, a substantial effort has been made for the application of textual data compression techniques to various computational biology tasks, ranging from storage and indexing of large datasets to comparison of genomic databases. In this paper we review the ways in which ideas and approaches fundamental to the theory and practice of data compression have been used in the area of bioinformatics. We look at how basic theoretical ideas from data compression, such as Need of compression, lossy vs lossless compression, how to measure compression ratio, how do can we compress a biological sequence and why other universal text compression algorithms are not suitable for biological sequence.This article introduces several algorithms which are used in past twenty year for DNA sequence compression. These algorithms used enhance technique related to previous one with improved result. This paper introduced idea of popular DNA sequence compression algorithm such as Biocompress, Cfact Rival, Gen Compress, CTW-LZ, Genome Compress, DNACompress, Huff bit Compress, Hash based, look Up Table, Differential Direct Coding and LSDB method.

Keywords: DNA compression, Genome compression, compression, biocompression

I. INTRODUCTION

The human genome contains around 3 billion characters over 23 pairs of chromosomes. So sizes of the databases are very huge and complex, and hence to store, access and process this data efficiently is a difficult task. So it needs an efficient compression algorithm to store these huge mass of data. DNA (Deoxyribonucleic Acid) is the chemical name of molecules that carries genetic instructions in all living things. DNA consists of a chain made from four types of nucleotide subunits, each composed of: a five-carbon sugar (2'-deoxyribose), a phosphate group, and one of the four bases adenine, cytosine, guanine, and thymine. Each sugar is attached with one four bases .The most common form of DNA in a cell is in a double helix structure, in which two individual DNA strands twist around each other in a right-handed spiral. Since DNA sequences contain four symbols A, T, G and C, the most efficient way to represent them would be using two bits for each symbol. However, if one applies standard compression software such as the Unix \compress" and \compact" or the MS-DOS archive programs \pkzip" and \arj", they all expand the file with more than 2 bits per base, although all these compression software are universal compression algorithms. These software are designed for text compression, while the regularities in DNA sequences are much subtler. The two bit encoding is efficient if the bases are randomly distributed in the sequence. But the life of an organism is non-random, hence DNA sequences which appear in a living organism are expected to be nonrandom and have some constraints. In other words, they should be compressible. Compression of DNA sequences is a very challenging task. This can be seen by the fact that no commercial file-compression program achieves any compression on benchmark DNA sequences we use in this paper. Several compression algorithms specialized for DNA sequences have been developed in the past ten years.
It is true that the compression of DNA sequence is a difficult task for general compression algorithms, but at the same time, from the viewpoint of compression theory it is an interesting subject for understanding the properties of various compression algorithms. Here we discuss about the methodology of compression methods in brief.

II. A BRIEF SURVEY ON COMPRESSION METHODS

BioCompress -1 (1993-1994) and BioCompress -2[7, 16] 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. The algorithm combines substitutional and statistical methods and lead to the highest compression of DNA. The results, although not satisfactory, gives insight to the necessary correlation between compression and comprehension of genetic sequence.

CFACT E.RIVAL 1996

Cfact[30] not only detects repeats in a text but select some of them according to their compressibility. Cfact work in two step process –First one is to locate repeated segment in a sequence and the second one is to measure their quantitative importance by the compression rate. It searches the longest exact Matching repeat using sux tree data structure in an entire sequence. The idea of Cfact is basically the same as Biocompress-2 except that Cfact is a two-pass algorithm. It builds the sux tree in the first pass. In the encoding phase, the repetitions are coded with guaranteed gain; otherwise, two-bit per base encoding will be used. This is analogous to the secret word encoding condition in Biocompress-2 except that the order-2 arithmetic coding is not used in Cfact

GenCompress

Another algorithm is GenCompress[16] the performance of which is based on reference sequence selection as approximate matching with edit operations uses this reference sequence for compression. It take use of both approximate repeats and repeal 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.

CTW-LZ MASTUMORO 2000

The compression ratio achieved by the Context-Tree Weighting (CTW) algorithm is among the best possible and it certainly outperforms the Lempel-Ziv (LZ)[6,12] type of methods. The CTW algorithm actually performs a sophisticated model and parameter estimation and thus it can be used in other (data-mining) applications. In applications like these we only need the modeler (encoder) and therefore it will be useful to consider methods that will speed up the encoding process. The Context-Tree Weighting achieves a very good compression ratio but has a much slower execution speed than e.g. Lempel-Ziv type of algorithms.

GenomeCompress

The next algorithm GenomeCompress (U.Ghoshastide et al, 2005)[5,14,26] 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 25=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.

DNACompress

The next DNACompress(XIN Chen et al, 2002)[7,8,16] 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 encoding, 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 Pattern Hunter finishes up with its task, it is then determined which repeats should be worked upon to get 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.

HUFFBIT COMPRESS

HUFFBIT COMPRESS[9] uses the concept of extended binary trees for compression; assigns a zero and one for left and right child respectively. It is a two way process; in the initial phase it constructs an extended binary tree. In the best case the algorithm achieves a compression of 1.006 bits per base. Even though the algorithm is simple the compression rate achieved is not satisfactory and it needs to scan the entire sequence to obtain the frequency of individual bases before starting the compression procedure

HASHBASED

Another compression, which also divides the entirely scanned DNA sequence into factors of length four, is Hashbased (Ateet Mehta et al, 2010)[1]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.

LOOK UP TABLE

DNA sequence compression algorithm based on fixed length LUT[2,3,4,15,16] and LZ77[12](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

Differential Direct Coding (2D)[16,19] (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 pattern type like DNA or RNA. The presentation 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 content, which results in scrap data. None of them enables part by part sequence decompression.

LSDB METHOD

It stores the individual gene position in a compressed file. Since LSBD[13] method performs a gene based compression, further processing of compressed data reduces memory consumption. The biggest advantage of this algorithm is that it enables part by part decompression and can work on any data of any dimension. Here the method identifies individual gene location and then constructs triplets that are mapped to an eight bit number. The individual gene information is stored in a pointer table and a pointer are provided to corresponding location in the compressed file. The LSBD method correctly compresses the non-base characters and performs well on repeating sequences.

III. CONCLUSION

The size and importance of DNA sequences molecular biology databases will be bigger and bigger in the upcoming time ; therefore this information must be stored or communicated efficiently. Several different methods are used for DNA compression till now. In BioCompress the decompression degrades the algorithm performance. The GenCompress algorithm not works suitable when approximate repeat and approximate reverse generate errors. CTW achieve better compression ration but have slower execution rate than LZ algorithm. A Genome compression is very effective for storage and time complexity. GenomeCompress 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. DnaCompress uses 2 phase algorithm with PatternHunter tool which involve less searching time. DnaCompress encode only those repeats that provide maximum compression. HuffBit Compression uses concept of extended binary tree however unable to find satisfactory compression rate. In Hashbased it is not possible to process any part of sequence without decompression the entire sequence. Differential Direct Coded is unable to do compression part by part where the LSDB method enable part by part compression and can work on any sized of data. DNA sequence compression algorithm based on LUT and LZ77 includes a LUT based pre coding routine and LZ77 compression routine. . It includes a look up table describing mapping relationship between DNA segment and corresponding character. Three characters in DNA sequence are mapped into a character of 64 ASCII character. The Fixed length LUT works on fixed sized look up table. Its gives faster performance then previous algorithms in compression ratio and elapsed time. While the variable length LUT works on dynamic creation of LUT so it gains lower compression ratio and lesser memory consumption than fixed length LUT. So there is possibility of further improvement in these methods to achieve fast and effective DNA compression.

REFERENCES

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

[2]. 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.

[3]. 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.
[4]. 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.
[5]. U. Ghoshdastider, 2005, et al., "GenomeCompress: A Novel Algorithm for DNA Compression", ISSN 0973-6824.
[6]. Sheng Bao, 2005, et al. "A DNA Sequence Compression Algorithm Based on LUT and LZ77".
[7]. Xin Chen, 2002, et al.," DNA Compress: fast and effective DNA sequence Compression" BIOINFORMATICS APPLICATIONS NOTE, Vol. 18 no. 12, Pages 1696–1698.
[8]. X. Chen, M. Li, B. Ma, and J. Tromp, "Dnacompress:fast and effective dna sequence compression," Bioinformatics, vol. 18,2002.
[9]. P.Raja Rajeswari Dr. Allam Apparao Dr. R.Kiran Kumar "HUFFBIT COMPRESS – Algorithm to Compress DNA Sequences Using Extended Binary Trees." Journal of Theoretical and Applied Information Technology Page(s): 101-106 2005 – 2010
[10]. Choi Ping Paula Wu, 2008, et al., "Cross chromosomal similarity for DNA sequence compression", Bioinformatics 2(9): 412-416.
[11]. K.N. Mishra, 2010, "An efficient Horizontal and Vertical Method for Online DNA sequence Compression", IJCA (0975-8887), Vol3, PP 39-45.
[12]. J. Ziv and A. Lempel., "A universal algorithm for sequential data compression," IEEE Transactions on Information Theory, vol. IT-23, May 1977.
[13]. Choi Ping Paula Wu, 2008, et al., " Cross chromosomal similarity for DNA sequence compression", Bioinformatics 2(9): 412-416
[14]. Xin Chen, Sam Kwong and Ming Li, "A Compression Algorithm for DNA Sequences and Its Applications in Genome Comparison," Genome Informatics, Vol. 10, pp. 51-61, 1999
[15]. Ascii code. [Online]. Available: http://www.LookupTables.com/
[16]. Textual data compression in computational biology: a synopsis Raffaele Giancarlo∗ , Davide Scaturro and Filippo Utro, Vol. 25 no. 13 2009, pages 1575–1586 doi:10.1093/bioinformatics/btp117
[17]. Mridula, T.V.; Samuel, P., "Lossless Segment Based DNA Compression," Electronics Computer Technology (ICECT), 2011 3rd International Conference on , vol.2, no., pp.298-302, 8-10 April 2011
[18]. Bockhorst,J. and Jojic,N. (2007) Discovering patterns in biological sequences by optimal segmentation. In Proceedings of the 23rd Conference in Uncertainty in Artificial Intelligence. AUAI Press, in press.
[19]. Bolshoy,A. (2003) DNA sequence analysis linguistic tools: contrast vocabularies,compositional spectra and linguistic complexity. Appl. Bioinform., 2, 103–112.
[20]. Gregory Vey,2009, "Differential direct coding: a compression algorithm for nucleotide sequence data", Database, doi: 10.1093/database/bap013
[21]. Cao,M.D. et al. (2007) A simple statistical algorithm for biological sequence compression. In Proceedings of the IEEE Data Compression Conference (DCC). IEEE Computer Society, pp. 43–52.
[22]. Chor,B. and Tuller,T. (2007) Biological networks: comparison, conservation, and evolutionary via relative description length. J. Comput. Biol., 14, 817–834.
[23]. Ferragina,P. et al. (2007) Compression-based classification of biological sequences and structures via the Universal Similarity Metric: experimental assessment. BMC Bioinformatis, 8, 252.
[24]. Ferragina,P. et al. (2008) Compressed text indexes: From theory to practice. ACM J. Exp. Alg., 13.
[25]. Loewenstern,D. et al. (1995) DNA sequence classification using compression-based induction. Technical report, DIMACS.
[26]. Matsumoto,T. et al. (2000) Biological sequence compression algorithms. Genome Inform., 11, 43–52.
[27]. Nevill-Manning,C.G. and Witten,I.H. (1997) Compression and explanation using hierarchical grammars. Comput. J., 40, 103–116.
[28]. Kocsor,A. et al. (2005) Application of compression-based distance measures to protein sequence classification: a methodological study. Bioinformatics, 22, 407–412.
[29]. G. Korodi and I. TABUS, "An efficient normalized maximum likelihood algorithm for dna sequence compression," ACM Transactions on Information Systems, vol. 23, 2005.
[30]. Rivals, E., Delahaye, J.-P., Dauchet, M., and Delgrange, O., A Guaranteed Compression Scheme for Repetitive DNA Sequences, LIFL Lille I University, technical report IT-285, 1995.