Journal of Computer Science 9 (6): 690-698, 2013
ISSN: 1549-3636

doi:10.3844/jcssp.2013.690.698 Published Online 9 (6) 2013 (http://www.thescipub.com/jcs.toc)

A Deoxyribonucleic Acid Compression Algorithm Using Auto-Regression and Swarm Intelligence

[1]Walid Aly, [2]Basheer Yousif and [3]Bassem Zohdy

[1]College of Computing and Information Technology,
Arab Academy for Science, Technology and Maritime Transport, Alexandria, Egypt
[2]Faculty of Computers and Information, Cairo University, Cairo, Egypt
[3]College of Computing and Information Technology,
Arab Academy for Science, Technology and Maritime Transport, Cairo, Egypt

Received 2012-12-31, Revised 2013-04-17; Accepted 2013-05-30

ABSTRACT

DNA compression challenge has become a major task for many researchers as a result of exponential increase of produced DNA sequences in gene databases; in this research we attempt to solve the DNA compression challenge by developing a lossless compression algorithm. The proposed algorithm works in horizontal mode using a substitutional-statistical technique which is based on Auto Regression modeling (AR), the model parameters are determined using Particle Swarm Optimization (PSO). This algorithm is called Swarm Auto-Regression DNA Compression (SARDNAComp). SARDNAComp aims to reach higher compression ratio which make its application beneficial for both practical and functional aspects due to reduction of storage, retrieval, transmission costs and inferring structure and function of sequences from compression, SARDNAComp is tested on eleven benchmark DNA sequences and compared to current algorithms of DNA compression, the results showed that (SARDNAComp) outperform these algorithms.

Keywords: DNA Compression, Autoregression, Particle Swarm Optimization, Lossless Compression

1. INTRODUCTION

Deoxyribonucleic Acid (DNA) sequence is the basic blue print of human beings, as DNA contains all of the instructions for each and every function of cells that make up all living organisms (Rajarajeswari and Apparao, 2011).

DNA is composed only from four chemical bases: Adenine (A), Thymine (T), Guanine (G) and Cytosine (C). Human DNA consists of about 3 billion bases and more than 99% of those bases are the same in all people, the order of this base determines the information available for building an organism (Meyer, 2010).

Understanding and analyzing DNA is a very important task that could lead to more improvements and customization of medical treatment, therapy for human, discovering new drug solutions and disease diagnosis (Mehta and Patel, 2010; Li *et al*., 2012a). Compression will help in storage, retrieval, querying and transfer of sequence data via any medium, studying, analysis and comparison of genomes.

However, the advancement of DNA sequencers generates hundreds of millions of short reads (Deorowicz and Grabowski, 2011) that leads biologists to produce exponentially large amounts of biological data every day and storing them in special databases such as EMBL, GenBank and DDBJ (Kuruppu *et al.*, 2012).

The Compression of this huge amount of produced DNA sequences is a very important and challenging task (Mehta and Patel, 2010; Hossein *et al*., 2011), DNA sequences are not random which means the ability to be very compressible (Hossein *et al*., 2011).

General purpose compression algorithms expand the sequences rather than compressing (Rajarajeswari and Apparao, 2011), so they cannot achieve the same compression ratio as specialized DNA sequences compression algorithms.

Corresponding Author: Walid M. Aly, College of Computing and Information Technology, Arab Academy for Science, Technology and Maritime Transport, Alexandria, Egypt

As DNA sequences consists of four nucleotides bases, two bits should be enough to store each base, in spite of this fact, the standard compression algorithm like "compress", "gzip", "bzip2", "winzip" uses more than 2 bits per base (Mridula and Samuel, 2011).

1.1. DNA Sequences Compression Modes

Compression modes of DNA sequences could be categorized into two modes as shown in **Fig. 1**, these modes are:

- Horizontal mode
- Vertical mode

1.2. Horizontal Mode

Works through making use of information contained in a single sequence by making reference only to its bases, in this mode the compression works on sequences one by one, the typical methods of horizontal mode can be classified as follows:

- Substitutional-statistical combined methods which work by partitioning the sequence into substrings, some of partitioned substrings are compressed by substitutional methods and the remaining substrings are compressed by the statistical methods, substitutional technique was proposed by Storer and Szymanski and statistical technique was proposed by Thomas and Cover (Giancarlo *et al*., 2012)
- Transformational methods that relies on transforming the sequence before the compression takes places (Giancarlo *et al*., 2012)
- Grammar-based methods in which a text string is compressed by using a context-free grammar, then the string is encoded by a proper encoding of the relevant production rules (Giancarlo *et al*., 2012).

1.3. Vertical Mode

Works by using information contained in the entire set of sequences by making reference to the bases of the entire set of sequences (Giancarlo *et al*., 2012), different methods were introduced in vertical mode compression including table Compression, which introduced by (Kaipa *et al*., 2010).

The scope of this research is to propose a lossless DNA compression algorithm using substitutional-statistical methods in horizontal mode, this proposed algorithm relays on Autoregressive modeling (AR), optimized by Particle Swarm Optimization (PSO) to reach a satisfactory compression ratio, that will lead to better assistance for biologists and scientists in their research, storage and transfer of biological data.

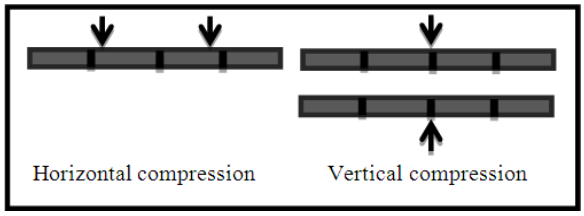

Fig. 1. Diagram shows horizontal and vertical compression

Research studies to solve the DNA compression problem is still in progress to develop a technique that reach satisfactory compression ratio. To the best of author's knowledge, this will be the first study that uses the auto-regressive modeling in compressing the DNA sequences aided with swarm intelligence. Since DNA sequences is a discrete sequence, so Autoregressive (AR) could be used for DNA sequence analysis, studying and comparison.

The study is organized as follows: Section two includes a review of a number of other specialized compression algorithms for DNA, Section three includes an introduction to AR, Section four present a brief introduction to PSO, Section five includes detailed illustration of our proposed algorithm and showing the results of applying our algorithm to eleven benchmark problems, followed by conclusion and future work.

2. RELATED WORK

Working in DNA compression was initially presented by Grumbach and Tahi in their pioneer work of DNA sequences compression by BioCompress Algorithm (Pinho *et al*., 2011) and its second version BioCompress-2, these algorithms are based on Ziv-Lempel compression technique (Berger and Mortensen, 2010), BioCompress-2 search for exact repeats in already encoded sequences, then encodes that repeats by repeat length and the position of preceding repeat appeared, when no repetition is found it uses order-2 arithmetic coding (Lin *et al*., 2009).

The Cfact technique (Merino *et al*., 2009) searches for the most lengthy exact match repeat, then uses a suffix tree on the entire sequence, by two passes, repeats are then encoded when gain is guaranteed, or using two bits per base for encoding.

GenCompress algorithm (Claude *et al*., 2010) released with two versions GenCompress-1 and Gencompress-2, in the first release the algorithm uses the technique of hamming distance or substitution only for the repeats, while GenCompress-2 uses deletion, insertion and substitution to encode repeats.

CTW+LZ (Kuruppu *et al*., 2012) uses context tree weighting combining LZ-77 type method, the algorithm

encodes lengthy approximate repeats by LZ-77, the short repeats are then encoded by CTW, but the execution time is very high for lengthy sequences.

DNACompress (Grassi *et al*., 2012), employs Ziv-Lampel compression, it has two phases, in first phase it search and finds approximate repeats using software named Pattern Hunter, then encoding the repeated and non-repeated fragments, this algorithm have less execution time than GenCompress.

The DNAC (Kurniawan *et al*., 2009) algorithm compress the DNA sequence in four phases, at first phase it builds a suffix tree to find the exact repeats, in the second phase the exact repeats extends into approximate repeats through dynamic programming, the third phase it elicits the optimal non-overlapping repeats from the overlapped ones, in final phase the sequence is encoded.

DNASequitur (Lin and Li, 2010) algorithm is a grammar-based compression algorithm that deduces a context-free grammar to show the input data.

DNAPack algorithm (Kuruppu *et al*., 2012) uses hamming distance for repeats and CTW or Arth-2 compression for non-repeat regions, these algorithms performs well than other algorithms in this time period as it uses dynamic programming method in selection of repeat regions.

XM (Kaipa *et al*., 2010) is a statistical compression algorithm that calculates the probability distribution of each nucleotide using a set of experts namely: order-2 markov models, order-1 context markov models and copy expert that consider the next nucleotide as a part of copied region, then the results of experts are combined and sent to arithmetic encoder.

GRS compression algorithm (Wang and Zhang, 2011) is applied by compressing a sequence based on another sequence as a reference without dealing with any other information about those sequences.

DNABIT (Rajarajeswari and Apparao, 2011) has two phases, first even bit technique which assigns two bits for every nucleotide of non-repeat regions; second phase is odd bit technique which assigns 3, 5, 7 or 9 bits based on the size of repeat regions.

CDNA (Wu *et al*., 2010) and ARM (Pique-Regi *et al*., 2012) algorithms calculate the probability distribution of each symbol that optioned by approximate partial matches which having a small hamming distance to the context before the symbol that could be encoded. The ARM algorithm is concerned of how the sequence is generated by calculating the probability of the sequence (Gupta *et al*., 2010).

DNAZip (Gupta *et al*., 2010) have two phases, the first phase is a transformation that is applied to the sequence and the second phase is concerned with encoding the transformed sequence.

The algorithms proposed by (Makinen *et al*., 2010) compresses not only the related sequences but also have retrieval functionality that returns the substring from its position in sequence and returns the number of occurrences of substring and return the position when the substring occurs in a collection.

Another algorithm proposed by (Bharti and Singh, 2011) that process in two phases, in first phase a shell search is done for specific length of palindromes which is three bases this is done by checking all possible places in the sequence, the algorithm core process is processed by comparing the first base from the sequence with the first letter from the end of the sequence and the second from the beginning with the second from the end and then the algorithm print the output when a palindrome is correlated in some way.

3. AUTOREGRESSIVE MODELING

A DNA sequence contains repeats that could be exact or approximate. Bases within each sequence could be repeated in some form of a model that could lead to better studying and analysis of sequences.

As DNA sequences is a discrete sequence, techniques like AR could be used and applied to that sequences, it's remarkable that AR model was recognized as an efficient tool to the coding of DNA sequences (Yu and Yan, 2011).

AR used to model and predicts various types of natural phenomena and it is one of the group of linear prediction formulas that attempt to predict an output of a system based on the previous outputs. Since correlations have been related to biological properties of the DNA, AR modeling could be used to model it.

In linear prediction analysis, a sample in a numerical sequence (the bases in the DNA sequence are represented as numbers as will be illustrated further) is approximated by linear combination of either preceding or future values of the sequence (Yu and Yan, 2011).

Forward Linear Prediction given by Equation 1:

$$e(n) = x(n) - a_1 x(n-1) - a_2 x(n-2) - .. - a_p x(n-p) \quad (1)$$

Where:

x = Numerical sequence,
n = Current sample index,
$a_1, a_2 \ldots a_p$ = linear prediction parameters,

To apply the AR modeling on DNA sequences, basically the sequence must be transformed to numbers by assigning a numeric values to nucleotides of the sequences to facilitate the calculation of AR parameters, in this study the AR is used to predict the nucleotides of

the DNA sequences, linear prediction parameters are determined using PSO.

4. PARTICLE SWARM OPTIMIZATION

Modeling of swarms was initially proposed by Kennedy to simulate the social behavior of fish and birds, the optimization algorithm was presented as an optimization technique in 1995 by Kennedy and Eberhart (Eslami *et al*., 2012), PSO has particles which represent candidate solutions of the problem, each particle searches for optimal solution in the search space, each particle or candidate solution has a position and velocity.

A particle updates its velocity and position based on its inertia, own experience and gained knowledge from other particles in the swarm, aiming to find the optimal solution of the problem.

The particles update its position and velocity according to the following Equation 2:

$$v_i^{k+1} = wv_i^k + c_1 rand_2 \times (pbest_i - s_i^k) + (gbest_i - s_i^k) \quad (2)$$

Where:

v_i^{k+1} = Velocity of agent i at iteration k,
w = Weighting function,
c_j = Weighting factor,
rand = Random number between 0 and 1,
S_i^k = Current position of agent i at iteration k,
$pbest_i$ = Pbest of agent i,
$gbest_i$ = gbest of the group.

The weighting function used in Equation 1:

$$w = w_{max} - \frac{w_{max} - w_{min}}{iter_{max}} \times iter \quad (3)$$

Where:

W_{max} = Initial weight
W_{min} = Final weight
$iter_{max}$ = Maximum iteration number
iter = Current iteration number

According to (Sedighizadeh and Masehian, 2009) more than ninety modification are applied to original PSO, in this research the original PSO with dynamic weighting factor is applied to solve the optimization problem of the compression of DNA sequences using AR by determining the linear prediction coefficients, since these coefficients of the AR are numbers between 0 and 1, the PSO role here is to optimize the coefficients to reach maximum compression rate.

5. PROPOSED WORK: SARDNACOMP

This research proposes a Swarm Auto-Regression DNA Compression (SARDNAComp) algorithm.

The goal of SARDNAComp is to solve the DNA compression challenge by reaching higher compression ratio. The algorithm uses the AR to predict the bases based on previous four bases according to the following Equation 4:

$$Y(K) = A_0 + A_1 * Y_{(k-1)} + A_2 * Y_{(k-2)} + A_3 * Y_{(k-3)} + A_4 * Y_{(k-4)} \quad (4)$$

Where:

Y(k) = Base to be predicted at index k
A_0, A_1, A_2, A_3, A_4 = Random coefficients between 0 and 1

PSO is used to estimate the parameters of AR, based on its characteristics of benefiting of cognitive and social behavior between particles-which represent candidate solutions-PSO is considered to be a very efficient technique for estimating the parameters of AR (Wachowiak *et al*., 2012).

The implementation of PSO within SARDNAComp uses dynamic inertia weight as illustrated in Equation 3, this leads to enhancement of the precision and tuning the convergence of particles without trapping in a local minima point (Li *et al*., 2012b). The basic structure diagram for SARDNAComp is shown in **Fig. 2**.

The application of SARDNAComp can be illustrated in 6 steps, where the first three steps are for preparing the DNA sequence data for AR modeling.

5.1. SARDNAComp Steps

Step1: Read the DNA sequence file.
Step2: Reshape the sequence as 5 columns.
Step3: Assigning a numeric values to the bases (A,C,G and T) as 0.25,0.5,0.75,1 respectively, since this algorithm lies in the domain of statistical methods, the values of nucleotides must not exceed 1 before applying the AR for each row of the reshaped sequence.
Step4: PSO algorithm is applied to optimize the coefficients of the AR, each particle in each iteration will represent the coefficients in the AR model, coefficients represented by each particle will be used to build a model of its own, the AR equation is applied for each row of the sequence and fitness is calculated as in equation:

$$Fitness = \frac{Number\ of\ correct\ prdicted\ bases}{Total\ bases\ number\ of\ sequence} * 100$$

The output of this step will be the particle with the highest fitness and thus the best model is declared **Table 1** shows the tuning parameters of SARDNAComp:

Step5: Compare the result of sequence produced by the AR model for the predicted nucleotide, if it is correct then the nucleotide is removed from the sequence, if not, it remains in the sequence.

Step6: The algorithm outputs-i-Flag file contains a series of ones and zeros as an indication whether each nucleotide as modeled correctly or not:

- Coefficients of model
- DNA data in sequence that could not be modeled correctly

To verify the validity of compression algorithm a decompression algorithm is also developed, the inputs will be the outputs of compression algorithm which are the compressed file which contains only the unpredicted nucleotides, coefficients used and the flag file, the file of nucleotides is reshaped with blanks for predicted nucleotides, the running the AR equation for each nucleotide to retrieve it.

The PSO algorithm as mentioned, optimizes the coefficients of AR to choose the best coefficients that help the AR equation to better predict the nucleotides of the DNA sequence to assist in increasing the compression ratio, sample of a particle created by PSO presented in **Fig. 4**, the procedure of PSO algorithm starts with determining the objective function which is a function created and contains the AR formula, after that the algorithm initialize the variables which are the population size, here in this study the population size is set to 10 and dimensions is set to 5 since that we need 5 coefficients, maximum number of iterations is set to 100 and the cognitive and social parameters are set to 2 according to (Eslami *et al*., 2012), in the next step the swarm initializes and velocities, after the initialization the algorithm evaluate the initial population, then initializing local best for each particle and then finding the best particle in initial population, then starting the iterations which in turn updates the velocities and positions, then evaluating the new swarm and updating the local position for each particle, then the PSO transfers the best solution (Coefficients) to the AR function which in turn be applied on each row of the sequence. The global best of the PSO through iterations presented in **Fig. 3**.

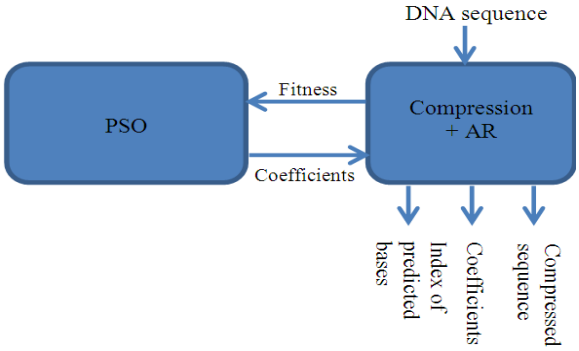

Fig. 2. SARDNAComp structure

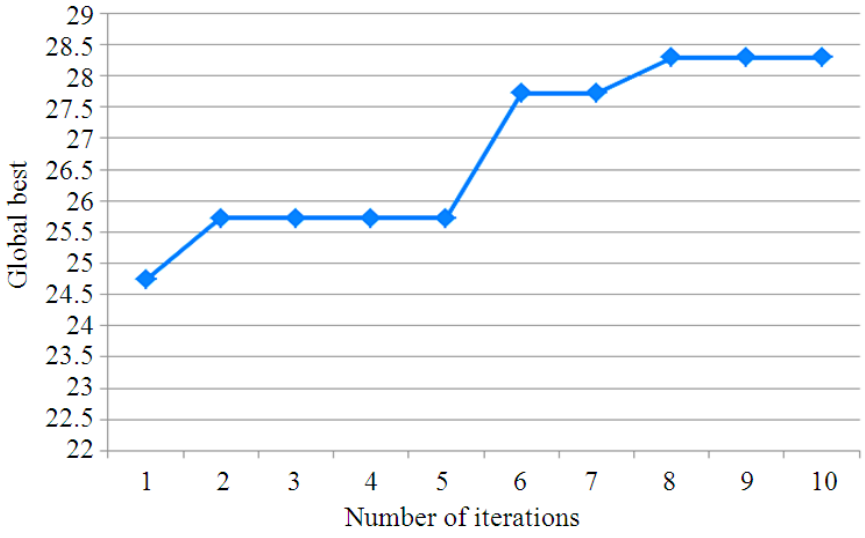

Fig. 3. Example of global best through iterations

Fig. 4. Sample of a particle

Table 1. PSO arguments

Population size	10
No. of particles	10
No. of generations	20
C1	2
C2	2
Inertia Weight	Adaptive

Algorithm (1): Procedure of Compression (SARDNAComp)

Inputs: DNA Sequence file
Outputs: Compressed Sequence File
Step(1): Read DNA Sequence File
Step(2): Reshaping the sequence file to 5 columns matrix
Step(3): Assigning nucleotides values:
For each nucleotide
A = 0.25, C = 0.5, G = 0.75, T = 1;
End
Step (4): PSO Algorithm
Step(5): Compare the results of AR with the original nucleotides.
For each row
If AR result = 5th nucleotide
Flag = 1;
Remove the base;
Else
Flag = 0;
Base remains in sequence;
End
End
Step(6): Output compressed DNA sequence file, flag file with index of predicted nucleotides, coefficients.

Algorithm (2): Procedure of PSO

Step(1): Initializing: popsize, Maxit, npar, c1,c2, constriction factor.
Step(2): Initializing Swarm and velocities:
Random(Population);
Random(Velocities);
Step(3):Evaluate initial population:
Calculate population cost;
Step (4): Initialize local best for each particle:
Location of local best;
Cost of local best;
Step(5): Finding Local best in initial population.
Step(6): Start iterations:
while iter < maxit
iter= iter + 1
Update velocity;
Update particles positions;
Evaluate the new swarm;
Update the best local position for each particle and the global best;
End while
Step(6): Calculate the fitness based on AR model and fitness function

Algorithm (3): Procedure of Decompression

Inputs: Compressed DNA Sequence file, index flag file and coefficients used in compression
Outputs: Decompressed Sequence File
Step1: Reshape DNA compressed file as one vector array
Step2: Apply index file to know the position of each predicted nucleotide
Step3: Run AR for each blank and replace it with the original nucleotide.
Step4: Compare results with original file.

6. RESULTS

SARDNAComp applied on eleven benchmark DNA sequences. In this study for the best cases it takes 1.333 bits per base as a compression ratio. The results shows that the proposed algorithm (SARDNAComp) achieves the best compression ratio among all other algorithms, the compression algorithm is developed by MATLAB version 7.6.0 (R2008), on a Core 2 Due processor with a 3 GB of RAM, both the compression ratio or the compression time considered to be outstanding to the best of author's knowledge. The compression rate of each sequence of the eleven benchmark sequences is presented in **Table 2**, the mean bits per base of the algorithms on DNA sequences is illustrated in **Fig. 5**, a comparison of sequences before compression and after being decompressed by SARDNAComp presented in **Fig. 6** that shows that the conditions after and before are similar.

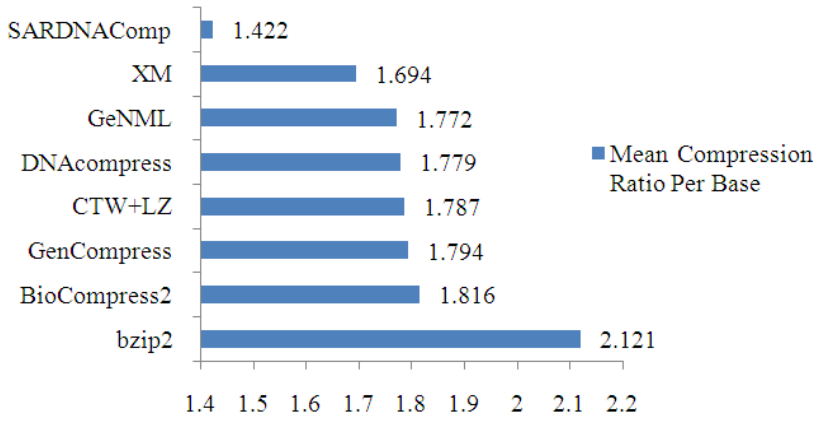

Fig. 5. Mean bits per base for DNA compression algorithms

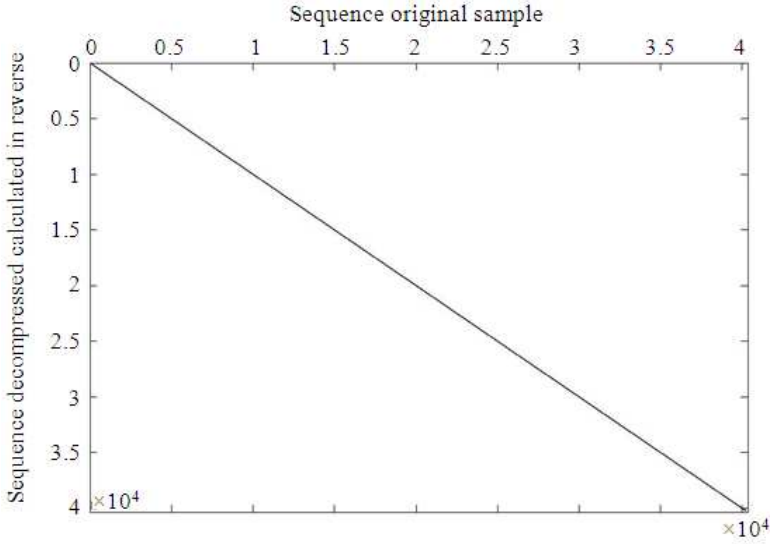

Fig. 6. Comparison of original DNA sequence files before and after compression

Table 2. Bits per base for the eleven benchmark DNA sequences after compression

Sequence	Length (bases)	Bits for bases (before compression)	Bits for bases (after compression)	Ratio (before compression)
CHMPXX	121024	242048	161710	1.336
CHNTXX	155943	311886	223768	1.434
HEHCMVCG	229354	458708	333854	1.455
HUMDYSTROP	38770	77540	57520	1.483
HUMGHCSA	66495	132990	95458	1.435
HUMBB	73308	146616	105394	1.437
HUMHDABCD	58864	117728	84416	1.438
HUMHPRTB	56737	113474	83136	1.465
MPOMTCG	186609	373218	264844	1.419
SCCHRIII	316613	633226	448380	1.416
VACCG	191737	383474	255694	1.333

7. CONCLUSION

In this research a new compression Algorithm proposed (SARDNAComp) for solving the DNA sequence compression problem, using the Autoregression (AR) modeling and optimized by particle swarm optimization to optimize the coefficients used in the AR that applied on the nucleotides of sequences after converting it to numeric values, the technique lies in the domain of statistical-substitutional method, the algorithm is applied on eleven benchmark DNA sequences and compared with other compression algorithms, the results shows that the compression ratio is superb among other algorithms.

The future work would concentrate on developing the autoregressive modeling to increase and reach better compression ratios, developing a substitution technique for the sequence that the Autoregression modeling couldn't predict and merge the two techniques for higher compression ratio. Publishing the application online to serve wide range of researchers. Also developing the application by allowing to compress the compressed sequence multiple times to reach higher compression ratio.

8. REFERENCES

Berger, M.S. and B.B. Mortensen, 2010. Fast pattern matching in compressed data packages. Proceedings of the IEEE GLOBECOM Workshops, Dec. 6-10, IEEE Xplore Press, Miami FL., pp: 1591-1595. DOI: 10.1109/GLOCOMW.2010.5700208

Bharti, K.B. and R.K. Singh, 2011. Biological sequence compression based on complementary palindrome using variable length Look Up Table (LUT). Int. J. Res. Rev. Applied Sci., 9: 461-463.

Claude, F., A. Farina, M.A. Martinez-Prieto and G. Navarro, 2010. Compressed q-gram indexing for highly repetitive biological sequences. Proceedings of the IEEE International Conference on BioInformatics and BioEngineering, May 31-Jun. 3, IEEE Xplore Press, Philadelphia, PA., pp: 86-91. DOI: 10.1109/BIBE.2010.22

Deorowicz, S. and S. Grabowskim, 2011. Compression of DNA sequence reads in FASTQ format. Bioinformatics, 27: 860-862. DOI: 10.1093/bioinformatics/btr014

Eslami, M., H. Shareef, M. Khajehzadeh and A. Mohamed, 2012. A survey of the state of the art in particle swarm optimization. Res. J. Applied Sci., 4: 1181-1197.

Giancarlo, R., D. Scaturro and F. Utro, 2012. Textual data compression in computational biology: Algorithmic techniques. Comput. Sci. Rev., 6: 1-25. DOI: 10.1016/j.cosrev.2011.11.001

Grassi, E., F.D. Gregorio, I. Molineris, 2012. KungFQ: A simple and powerful approach to compress fastq files. IEEE/ACM Trans. Comput. Biol. Bioinform., 9: 1837-1842. DOI: 10.1109/TCBB.2012.123

Gupta, A., V. Rishniwal and S. Agarwal, 2010. Efficient storage of massive biological sequences in compact form. Proceedings of the 3rd International Conference, Aug. 9-11, IEEE Xplore Press, Noida, pp: 13-22. DOI: 10.1007/978-3-642-14825-5_2

Hossein, M.S., P. Maiti and A. Mukherjee, 2011. Online genome compression software. IJCTE, 3: 141-147. DOI: 10.7763/IJCTE/IJCTE.2011.V3.296

Kaipa, K.K., A.S. Bopardikar, S. Abhilash, P. Venkataraman and K. Lee et al., 2010. Algorithm for DNA sequence compression based on prediction of mismatch bases and repeat location. Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine Workshops, Dec. 18-18, IEEE Xplore Press, Hong, Kong, pp: 851-852. DOI: 10.1109/BIBMW.2010.5703941

Kurniawan, T.B., N.K. Khalid, Z. Ibrahim, M.S.Z. Abidin and M. Khalid, 2009. Sequence design for direct-proportional length-based DNA computing using population-based ant colony optimization. Proceedings of the ICCAS-SICE, Aug. 18-21, IEEE Xplore Press, Fukuoka, pp: 1486-1491.

Kuruppu, S., B. Beresford-Smith, T. Conway and J. Zobel, 2012. Iterative dictionary construction for compression of large DNA data sets. Proceedings of the IEEE/ACM Transactions on Computational Biology and Bioinformatics, IEEE Xplore Press, pp: 137-149. DOI: 10.1109/TCBB.2011.82

Li, C., Z. Ji and F., Gu, 2012a. Efficient parallel design for BWT-based DNA sequences data multi-compression algorithm. Proceedings of the International Conference on Automatic Control and Artificial Intelligence, Mar. 3-5, IEEE Xplore Press, Xiamen, pp: 967-970. DOI: 10.1049/cp.2012.1137

Li, C., S., Yang, T.T., Nguyen, 2012b. A self-learning particle swarm optimizer for global optimization problems. IEEE Trans. Syst. Man Cybernet., 627-646. DOI: 10.1109/TSMCB.2011.2171946

Lin, J. and Y. Li, 2010. Finding approximate frequent patterns in streaming medical data. Proceedings of the IEEE 23rd International Symposium on Computer-Based Medical Systems, Oct. 12-15, IEEE Xplore Press, Perth, WA., pp: 13-18. DOI: 10.1109/CBMS.2010.6042675

Lin, P., L. Shaopeng, Z. Ixia and H. Peijie, 2009. Compressed pattern matching in DNA sequences using multithreaded technology. Proceedings of the 3rd International Conference on Bioinformatics and Biomedical Engineering, Jun. 11-13, IEEE Xplore Press, Beijing, pp: 1-4. DOI: 10.1109/ICBBE.2009.5162550

Makinen, V., N. Gonzalo, S. Jouni and V. Niko, 2010. Storage and retrieval of highly repetitive sequence collections. J. Comput. Biol.. 17: 281-308. DOI: 10.1089/cmb.2009.0169.

Mehta, A. and B. Patel, 2010. DNA compression using hash based data structure. Int. J. Inform. Technol. Knowl. Manage., 2: 383-386.

Merino, S., A. Retolaza, V. Trabadelo, A. Cruz and P. Heredia *et al*., 2009. Protein patterning on the micro- and nanoscale by thermal nanoimprint lithography on a new functionalized copolymer. J. Vacuum Sci. Technol., 6: 2439-2443. DOI: 10.1116/1.3264687

Meyer, S., 2010. Signature in the Cell: DNA and the Evidence for Intelligent Design. 1st Edn., HarperOne, ISBN-10: 0061472794, pp: 55.

Mridula, T.V. and P. Samuel, 2011. Lossless segment based DNA compression. Proceedings of the 3rd International Conference on Electronics Computer Technology, Apr. 8-10, IEEE Xplore Press, pp: 298-302. DOI: 10.1109/ICECTECH.2011.5941705

Pinho, A.J., D. Pratas, and P.J.S.G. Ferreira, 2011. Bacteria DNA sequence compression using a mixture of finite-context models. Proceedings of the IEEE Statistical Signal Processing Workshop, Jun. 28-30, IEEE Xplore Press, pp: 125-128. DOI: 10.1109/SSP.2011.5967637

Pique-Regi, R., A. Ortega, A. Tewfik and S. Asgharzadeh, 2012. Detecting changes in DNA copy number: Reviewing signal processing techniques. IEEE Signal Process. Mag., 29: 98-107. DOI: 10.1109/MSP.2011.943010

Rajarajeswari, P. and A. Apparao, 2011. DNABIT compress-genome compression algorithm. Bioinformation, 5: 350-360. PMID: 21383923

Sedighizadeh, D. and E. Masehian, 2009. Particle swarm optimization methods, taxonomy and applications. Int. J. Comput. Theory Eng., 1: 486-502. DOI: 10.7763/IJCTE.2009.V1.80

Wachowiak, M.P., R. Wachowiak-Smolikova and J. Zimmerling, 2012. The viability of global optimization for parameter estimation in spatial econometrics models. Department of Computer Science and Mathematics, Nipissing University.

Wang, C. and D. Zhang, 2011. A novel compression tool for efficient storage of genome resequencing data. Nucl. Acid Res., 39: 45-45. DOI: 10.1093/nar/gkr009

Wu, G.F., Y.L. Hou, W.R. Hou, Y. Song and T. Zhang, 2010. cDNA, genomic sequence cloning and sequences analysis of the fau gene from the Giant panda (Ailuropoda melanoleuca). Proceedings of the 4th International Conference on Bioinformatics and Biomedical Engineering, Jun. 18-20, IEEE Xplore Press, Chengdu, pp: 1-6. DOI: 10.1109/ICBBE.2010.5517214

Yu, S. and H., Yan, 2011. Correlated structural features and their applications to exon recognition in DNA seqeunces. Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Oct. 9-121, IEEE Xplore Press, pp: 3064-3069. DOI: 10.1109/ICSMC.2011.6084130