A DEOXYRIBONUCLEIC ACID COMPRESSION ALGORITHM USING AUTO-REGRESSION AND SWARM INTELLIGENCE

Walid Aly, Basheer Yousif and Bassem Zohdy

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

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 blueprint 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 consist 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. 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 place (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 autoregressive 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-Lempel compression, it has two phases, in first phase it searches 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

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:

\[ v_{i}^{k+1} = w v_{i}^{k} + c_{1} \text{rand} \times (p_{best} - s_{i}^{k}) + (g_{best} - s_{i}^{k}) \]  

(2)

Where:
- \( v_{i}^{k+1} \) = Velocity of agent i at iteration k,
- \( w \) = Weighting function,
- \( c_{1} \) = Weighting factor,
- \( \text{rand} \) = Random number between 0 and 1,
- \( s_{i}^{k} \) = Current position of agent i at iteration k,
- \( p_{best} \) = Pbest of agent i,
- \( g_{best} \) = gbest of the group.

The weighting function used in Equation 1:

\[ w = w_{\text{max}} - \frac{w_{\text{max}} - w_{\text{min}}}{\text{iter}_{\text{max}}} \times \text{iter} \]  

(3)

Where:
- \( w_{\text{max}} \) = Initial weight
- \( w_{\text{min}} \) = Final weight
- \( \text{iter}_{\text{max}} \) = Maximum iteration number
- \( \text{iter} \) = 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:

\[ Y(k) = A_0 + A_1 Y_{(k-1)} + A_2 Y_{(k-2)} + A_3 Y_{(k-3)} + A_4 Y_{(k-4)} \]  

(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:

\[ \text{Fitness} = \frac{\text{Number of correct predicted bases}}{\text{Total bases number of sequence}} \times 100 \]
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:

Step 5: 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.

Step 6: 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 a particle created by 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. 4. Sample of a particle

Table 1. PSO arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size</td>
<td>10</td>
</tr>
<tr>
<td>No. of particles</td>
<td>10</td>
</tr>
<tr>
<td>No. of generations</td>
<td>20</td>
</tr>
<tr>
<td>C1</td>
<td>2</td>
</tr>
<tr>
<td>C2</td>
<td>2</td>
</tr>
<tr>
<td>Inertia Weight</td>
<td>Adaptive</td>
</tr>
</tbody>
</table>

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

![Graph showing mean bits per base for DNA compression algorithms.]

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

![Graph showing comparison of original DNA sequence files before and after compression.]

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

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


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


