International Journal of Computer Theory and Engineering, Vol.3, No.1, February, 2011
1793-8201

Online Genome Compression Software

Md. Syed Mahamud Hossein[1], Pradip Maiti[2] and Arunima Mukherjee[3]

Abstract—**This is web based project which mainly deals with GENOMIC COMPRESSION. Here we have used several compression techniques i,e Huffman Compression Techniques, Four base to single base compression techniques..etc for compressing Nucleotide sequence of huge size. There are two phases one is ADNMINISTRATOR and another NORMAL USER. ADMINISTRATOR handles the data and maintains the database. Initially our aim to generate the encoded file for a particular file at runtime and the signature of that particular file are stored in another file to identify that particular file while decoding but due to stored of time we were not able to generate at runtime but rather we store the encoded file along with signature file in the database and while retrieving decoded data from encoded data we use encoded data file along with the signature file.The DNA sequences storing and transmitting them may require a huge amount of space. This web page are help to reduce the space for storing and transmitting data , also introduce one new techniques along with exiting Huffman Technique of compression routine. DNA and RNA sequences can be considered as tests over a four letter alphabet, namely {a,t,g and c}. This new algorithm can approach a compression rate of 2.1 bits /base and even lower. Time complexity of the algorithm O(n). The time complexity of this algorithm is linear and varies linearly with the size of the source file to be compressed. For accuracy purpose we have use mapping techniques in between input file and output file. The greatest advantage of this program is fast execution, small memory occupation and easy implementation.**

***Index Terms*—Biology, Data Compression, Data Decompression, Genetics**

I. Introduction

With more and more complete genomes of prokaryotes and eukaryotes becoming available and the completion of Human Genome Project on the horizon, fundamental questions regarding the characteristics of these sequences arise. Life represents order. It is not chaotic or random [2]. Thus, we expect the DNA sequences that encode life to be nonrandom. In other words, they should be very compressible. There is also strong biological evidence that supports this claim. It is well known that DNA sequences only consist of four nucleotide bases $\{a, c, g, t\}$, and one byte are enough to store each base. All this evidence gives more concrete support that the DNA sequences should be reasonably compressible. It is well recognized that the compression of DNA sequences is a very difficult task [3-9]. We have done a program to get back the DNA sequence in the original form before compression which we call as Decompression. Over here we have design the program for the compression and decompression.

The main aim of this project is a large sequence of a polynucleotide the memory required to store the data would be more and our aim is to reduce the space required by the data, also to keep the data intact after decompression and increase the transmission cost . Also to protect the original data and using lesser space as possible which can be done by compressing the data and is to provide lossless data at the time of the decompression. Another aim is to make run time less and aim is to protect data. This project is to design the logic easy to understand and to decrease the complexity

II. Method

A. Method-1

The first optimal code was developed by David Huffman. Before suggesting an algorithm for generating an optimal code, he pointed out that an optimal code has some characteristics which are summarized in the following theorem that uses a source alphabet S = $\{x_1 \ldots\ldots x_n)$, a set of associated probabilities p = $\{^{p}{}_{I} \ldots\ldots {}^{p}n\}$ and code-words $\{c_1 \ldots.. c_n\}$ with corresponding lengths $\{l_1, \ldots, ln\}$.

Huffman Algorithm( )

- For each letter create a tree with a single root node and order all trees according to the probability of letter occurrence; while more than one tree is left.
- Take the two trees t_l, t_2 with the lowest probabilities p_1,p_3 and create a tree with probability in its root equal to $p_I + p_2$ and with t_l and t_2 as its subtrees; associate 0 with each left branch and 1 with each right branch;
- Create a unique codeword for each letter by traversing the tree from the root to
- the leaf containing the probability corresponding to this letter and putting all encountered 0s and Is together;
- The resulting tree has a probability of 1 in its root.
- Huffman saw the structure resulting from application of his algorithm as a net of tributary rivers eventually flowing into a large river. He thought associating Is and Os with branches was analogous to "the placing of signs by a water-borne insect at each of these junctions as he journeys downstream" with right turn posts (marked with I) and left turn posts (marked with 0), which would allow the insect to return back to the starting point.

It should be noted that the algorithm is not deterministic in the sense of producing a unique tree because, for trees with equal probabilities in the roots, the algorithm does not prescribe their positions with respect to each other either at the beginning or during execution. If t_1 with probability $^{p}{}_1$ is in the sequence of trees and the new tree t_2 is created with $p_1 = p_2$, should t_2 be positioned to the left of t_l or to the right? Also, if there are three trees t_1,t_2 and t_3 with the same lowest probability in the entire sequence, which two trees should be chosen to create a new tree? There are three possibilities for choosing two trees. As a result, different trees can be obtained depending on where the trees with equal probabilities are placed in the sequence with respect to each other. Interestingly, however, regardless of the shape of the tree, the average length of codeword remains the same.

HUFFMAN CODING:

```
Data Structure used: Priority queue = Q
Huffman I
n = |c|
Q = c
for i =1 to n-1
   do z = Allocate-Node ()
   x = left[z] = EXTRACT_MIN(Q)
   y = right[z] = EXTRACT_MIN(Q)
   f[z] = f[x] + f[y]
   INSERT (Q, z)
   return EXTRACT_MIN(Q)
```

B. Method-2

The coding for the Compression and Decompression technique is design with the help of the “C”programming concept.

Consider a finite sequence*s* over the DNA alphabet {a, c, g, t}. An exact FBS in *s* that can be transformed from another substring in *s* with edit operations (conversion(Binary to decimal), insertion). We only encode those substring are exactly match with four bases that provide profits on overall compression.

Example :

Let s=atggtagtaatgtacatg……..(n-1)

The first substring is S_0 =atgg, equivalent decimal value is $=0X2^7+0X2^6+0X2^5+1X2^4+1X2^3+0X2^2+1X2^1+0X2^0=26$, place ASCII character of equivalent 26 of decimal value.

The maximum and minimum values of FBSs is of aaaa and cccc, result are shown in figure-I the value is 0 to 255 and vice versa, it depends on user choice, but not more than limit range from 0 to 255., because any type DNA sequences are the combination of four symbols only.

Encoding Algorithm:

Algorithm for 'atgc' sequence compression using substitution by binary bit strings:-

Input:1.source(Filename of source file to be compressed) and 2.target(filename of file where compressed file is to be stored).

Output: The compressed file the filename of which is specified by 'target'.

Data Structures used: buffer{4} which is an array of size 4.

Steps:

```
1. Open source file for reading and store the file pointer in variable 'fs':
fs=openfile(source)
2. Create file for storing the compressed file and store the file pointer in variable 'ft':
ft=createfile(target)
3. While end of file of source file is not reached do
   i. For i=1 to 4 do
      A.If end of file of source file is not reached then
         a.Read a character from source file and store it in variable 'ch' and store 'ch' in buffer[i]:
         buffer[i]=ch
         b.Increment file pointer 'fs' to next position in source file.
      else
         a.Goto step iii.
      B.Endif.
   ii. Endfor
   iii. Store value of i in variable 'size':
   size=i
   iv. Put sub=0 and j=7.
   v. For i=1 to size do
      A.If buffer[i]='a' then put hb=0 and lb=0.
      B.If buffer[i]='c' then put hb=0 and lb=1.
      C.If buffer[i]='g' then put hb=1 and lb=0.
      D.If buffer[i]='t' then put hb=1 and lb=1.
      E.sub=sub+hb*2^j.
      F.j=j-1.
      G.sub=sub+lb*2^j.
      H.j=j-1.
   vi. Endfor.
   vii. Write value of variable 'sub' into target file pointed by file pointer 'ft' and increment the position of file pointer 'ft' to next position of target file.
4. Endwhile.
5. Write value of variable 'size' into target file pointed by file pointer 'ft'.
6. Close the source and the target files:
   i. close(fs)
   ii. close(ft)
7.End.
```

Decoding algorithm:

Algorithm for decompression of binary bit string substitution compression technique:-

Input:1.source(Filename of source file to be decompressed) and 2.target(filename of file where decompressed file is to be stored).

Output: The decompressed file the filename of which is specified by 'target'.

Data Structures used:buffer[4] which is an array of size 4.

Steps:

```
1. Open source file for reading and store the file pointer in variable 'fs':
   fs=openfile(source)
2. Create file for storing the decompressed file and store the file pointer in variable 'ft':
   ft=createfile(target)
3.While end of file of source file is not reached do
   i. Read a character from source file and store it in variable 'ch'.
   ii. Increment file pointer 'fs' to next position in source file.
   iii. Read a character from source file and store it in variable 'chsize'.
   iv. Increment file pointer 'fs' to next position in source file.
   v. If end of file of source file is reached then
      A.Put size=chsize.
   vi. Else
      A.Put size=4.
      B.Decrement file pointer 'fs' to previous position in source file.
   vii. Endif.
   viii. For i=size to 1 step -1 do
```

```
A. If ch mod 2=0 then
a.ch=ch/2.
b. if ch mod 2=0 then put buffer[i]='a'.
c. else
d. put buffer[i]='g'.
e. endif.
B.Else
a. ch=ch/2.
b. if ch mod 2=0 then put buffer[i]='c'.
c. else
d. put buffer[i]='t'.
e. endif.
C.Endif.
D.ch=ch/2.
ix. Endfor.
x. For i=1 to size do
A.Write the value of buffer[i] into target file pointed by file pointer 'ft' and increment the position of file pointer 'ft' to next position of the target file.
xi.Endfor.
4. Endwhile.
5. Close the source and the target files:
i. close(fs).
ii. close(ft).
6. End.
```

C. *Algorithm for file mapping*

MATCHING-PERCENTAGE(String_1,String_2)

This function take two string as argument and return the percentage of matching. Here first we calculate the number of errors i.e. mismatch in between this two string. After that we can calculate percentage by this following formulae:

$$\text{Percentage}=\frac{\text{Frame\_size} - \text{Error\_no}}{\text{Frame\_size}} * 100$$

Here Frame_size means the length of the String.

ALGORITHM :

```
• Step1 : frame_size=LENGTH(String_1);
• Step2 : Repeat step 3 to 5 while String_1 is NULL.
• Step3:Index=MISMATCH-INDEX(String_1,String_2).
• Step4 : IF Index>Length(String_1)-1 then goto step 6.
• Step5 : IF Index=Length(String_1)-1
      then String_1=NULL.
    ELSE
  String_1=SUBSTRING(String_1,(Index+1)).
  String_2=SUBSTRING(String_2,(Index+1)).
• Step6 : Error_no=Error_no + 1.
• Step7:Percentage=((Frame_size-Error_no)/Frame_size)*100.
• Step8 : Return Percentage.
```

III. Introduction to Software Used

Front End Tools:

J2EE

J2EE technology and its component-based model simplify enterprise development and deployment. The J2EE platform manages the infrastructure and supports the web services to enable development of secure, robust and interoperable business applications.

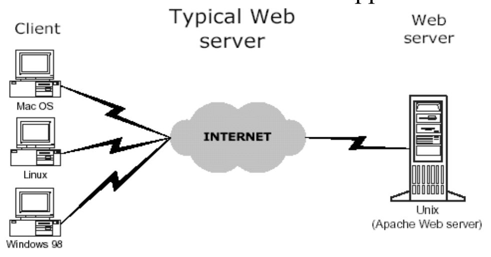
Developing on the Java platform means that projects are completed faster and with less debugging. Leveraging the J2EE [platform's] suite of technologies enable us to focus more of our technical resources on creating solutions to business problems. In addition, Java technology's underlying object oriented architecture allowed us to design our systems for maximum code reuse. We have to also take advantage of Java technology's portability by running the production application on both Windows NT and Solaris 8.0 Operating Environment with no code changes.

Recognizing that "one size doesn't fit all" Sun has grouped its innovative technologies into three editions: Java 2 Platform Micro Edition (J2ME Technology), Java 2 Platform Standard Edition (J2SE technology), and the Java 2 Platform Enterprise Edition (J2EE technology). Each edition is a developer treasure chest of tools and supplies that can be used with a particular product.

The J2EE platform is essentially a distributed application server environment. A Java Environment that provides the following:

A runtime infrastructure for hosting applications.

A set of Java extension API's to build applications.



Java Server Pages (JSP)

Java Server Pages (JSP) is a technology based on the Java language and enables the development of dynamic web sites. JSP was developed by Sun Microsystems to allow server side development. JSP files are HTML files with special Tags containing Java source code that provide the dynamic content. The following shows the Typical Web server, different clients connecting via the Internet to a Web server. In this example, the Web server is running on Unix and is the very popular Apache Web server.

First static web pages were displayed. Typically these were people's first experience with making web pages so consisted of My Home Page sites and company marketing information. Afterwards Perl and C were languages used on the web server to provide dynamic content. Soon most languages including Visual Basic, Delphi, C++ and Java could be used to write applications that provided dynamic content

Using data from text files or database requests. These were known as CGI server side applications. ASP was developed by Microsoft to allow HTML developers to easily provide dynamic content supported as standard by Microsoft's free Web Server, Internet Information Server (IIS). JSP is the equivalent from Sun Microsystems, a comparison of ASP and JSP will be presented in the following section.

The following diagram shows a web server that supports JSP files. Notice that the web server also is connected to a database.

JSP source code runs on the web server in the JSP Servlet Engine. The JSP Servlet engine dynamically generates the

HTML and sends the HTML output to the client's web browser.

JavaScript

JavaScript is the programming language of the web. JavaScripts can be inserted into HTML documents, to make the web pages more dynamic and interactive JavaScript is a scripting language. JavaScript is supported by Internet Explorer. When a JavaScript is inserted into an HTML document, the Internet browser will read the HTML and interpret the JavaScript.JavaScript gives HTML designers a programming tool JavaScript can put dynamic text into an HTML page Can write a variable text into display of an HTML page just like static HTML does. JavaScript can react to Events

A JavaScript can set to execute when some things happens, like when a page has finished loading or when a user clicks on an HTML element.

JavaScript can read and write HTML elements

A JavaScript can read an HTML element and change the content of an HTML element. JavaScript can be used to validate data in form before it is submitted to a server. The function is particularly well suited to save the server from extra Processing

JDBC-ODBC Connectivity

The JDBC-ODBC Connectivity is a JDBC driver that implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Connectivity implements JDBC for any database for which an ODBC driver is available. The Connectivity is implemented as the `sun.jdbc.odbc` Java package and contains a native library used to access ODBC.

Hyper Text Mark-Up Language (HTML)

Universal, non-proprietary, structured, text mark-up language Used to publish documents on the World Wide Web Used to define the structure of documents and links between documents An *application* of Standard Generalized Mark-up Language.

SGML is a meta-language, used to describe a mark-up language Latest SGML-based version of HTML is 4.01 Style sheets Scripting Internationalization

Accessibility

Feasibility Study:

Feasibility and Requirement Analysis

Introduction

After making a detailed investigation Feasibility study which is carried out to check the work ability of the proposed system. The Feasibility analysis is in a position to evaluate solution, second most studies tend to overlook the confusion inherent in system development the constraints and assured attributes, If Feasibility study is to serve as a decision document, it must answer some questions.

Feasibility Considerations

Feasibility study is conducted by considering mainly the following assumptions.

Economic Feasibility

An effective candidate system is evaluated by Economic analysis. It is also called as cost/benefit analysis, The procedure is to determine the benefits and savings that are expected from a candidate system and compare them that which cost should meet the users requirements, This effort that impress in accuracy in each phase of system life cycle. The proposed system is economically feasible as it lifetime project, there is one facility to understand that, in any department. Once it is installed that cannot be changed in between and is easy to understand.

Technical Feasibility

The Technical feasibility study centers on the existing computer system hardware and software etc., to what extent it can support the proposed addition. This eSeva proposed system is technically feasible too, it needs j2ee, oracle9i, NT based operating system, Web logic server, hence this is for a technical person to go through it, and easy to debug if in any case problem occurs.

Behavioral Feasibility

Computers are known to facilitate the changes, A strong estimation is made to react the user & likely to have towards the development of computerized system. Hence the introduction of a candidate system required a special effort to educate, sell and train the staff in number of ways for conducting the Business. This project needs little bit of computer knowledge to operate eSeva, the staff trained once to operate it. Since it all electronically processing system it has expenses during first time installation but it covers all money after it starts working. This is all pleasure of a Government that goes through eSeva.

System Design And Operational Environment

Introduction to System Design

System Design is the most important part of any Information System Development process. It specifies what the system should do to achieve the objective. User requirements are translated into design specification. System Design first involves the logical design and then physical constructions. Following steps are followed:

Detailed specifications of the system are established.

Feature of outputs, inputs, files, procedures that meet project requirement.

Data and procedures are linked together to produce the working system.

Logical Construction

It involves the description of detailed specification of procedures, control and security measures to be involved.

Physical Construction

It produces program software, files and working system. Programs are written as per the instruction of logical design phase that specifies what the system should do. Program accepts input from users, process data and produce reports and store data in files. A fundamental objective in the designing of information system is to ensure that it supports the business activity for which it is developed. To meet user it should accomplished the following: Performs right procedures properly. Present information and instructions in an acceptable and effective manner. Produces accurate results.

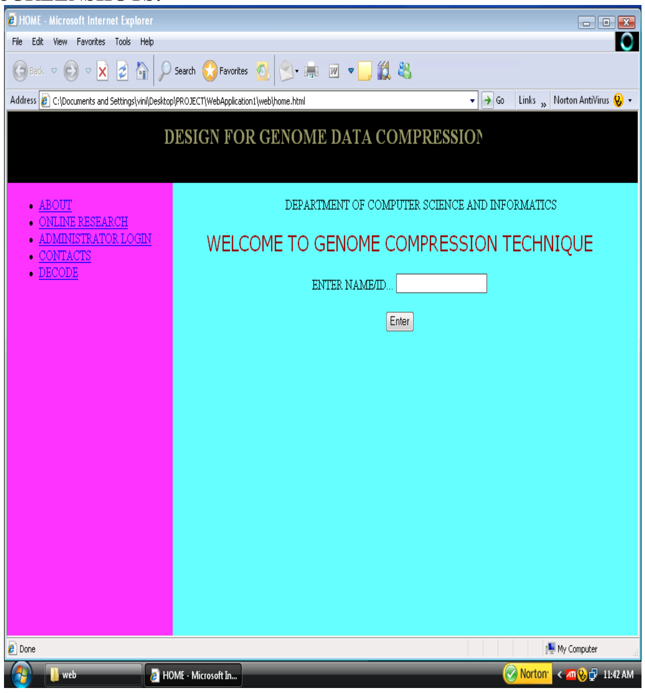
Provide an acceptable interface and method for interaction.

Input / Output Design

A major step in the system design is the preparation of input and the design of output reports in a form acceptable to the user. Without input data there is no system, so data must be provided in right form of input and information produced must be in a format acceptable to the use. These are described in detail in the following section.

In this process we have taken the genome sequence as input which consist of the Form A,T,G,C. alphabet. These sequence are compressed i.e 75% of size of the sequenced is reduced. The main advantage of this coding is that it takes less time at runtime. The compressed file is then converted into the ascii character which can be easily transmitted.
This ascii character file is required at the time of the decompression. The whole file is converted into the integer and then with the help of the modollar 4 power operator
We get the original data. The main advantage of this project is that we get loss less data. We can get 100% accurate result.

SCREENSHOTS:


HOME - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\Documents and Settings\vini\Desktop\PROJECT\WebApplication1\web\home.html
Enter
Done
My Computer

DESIGN FOR GENOME DATA COMPRESSIO

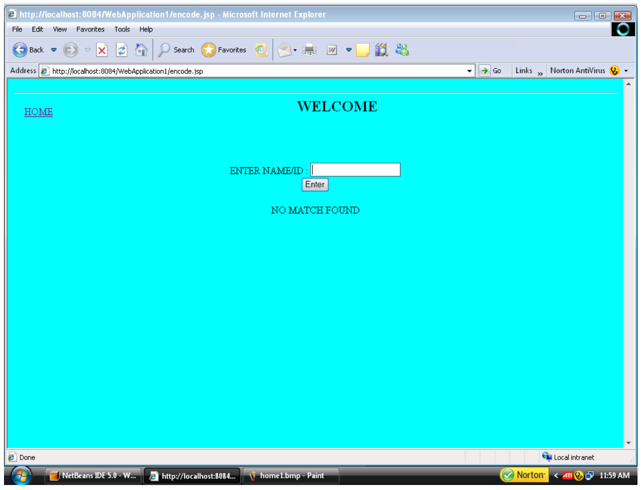
- ABOUT
- ONLINE RESEARCH
- ADMINISTRATOR LOGIN
- CONTACTS
- DECODE

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATICS

WELCOME TO GENOME COMPRESSION TECHNIQUE

ENTER NAME/ID...

HOME PAGE


http://localhost:8084/WebApplication1/encode.jsp - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://localhost:8084/WebApplication1/encode.jsp
Enter
Done
Local intranet

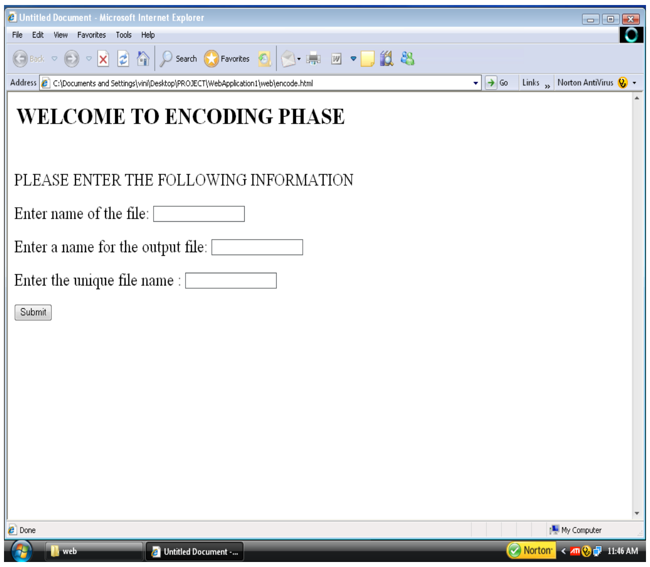
HOME

WELCOME

ENTER NAME/ID :

NO MATCH FOUND

COMPRESSION LOGIN


Untitled Document - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\Documents and Settings\vini\Desktop\PROJECT\WebApplication1\web\encode.html
Submit
Done
My Computer

WELCOME TO ENCODING PHASE

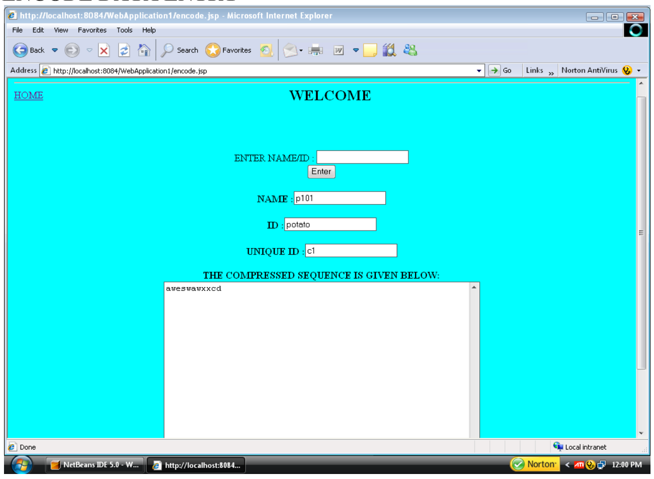
PLEASE ENTER THE FOLLOWING INFORMATION

Enter name of the file:

Enter a name for the output file:

Enter the unique file name :

ENCODE DATA ENTRY


http://localhost:8084/WebApplication1/encode.jsp - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://localhost:8084/WebApplication1/encode.jsp
Enter
Done
Local intranet

HOME

WELCOME

ENTER NAME/ID :

NAME : p101

ID : potato

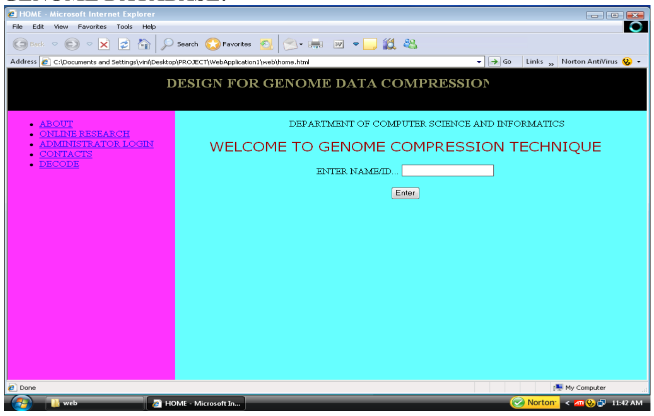
UNIQUE ID : c1

THE COMPRESSED SEQUENCE IS GIVEN BELOW:

```
avesvavxcd
```

ENCODED SEQUENCE

GENOME DATABASE:


HOME - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\Documents and Settings\vini\Desktop\PROJECT\WebApplication1\web\home.html
Enter
Done
My Computer

DESIGN FOR GENOME DATA COMPRESSIO

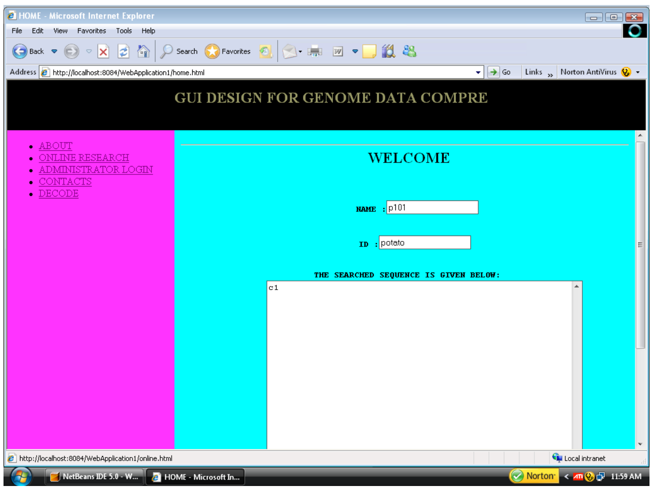
- ABOUT
- ONLINE RESEARCH
- ADMINISTRATOR LOGIN
- CONTACTS
- DECODE

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATICS

WELCOME TO GENOME COMPRESSION TECHNIQUE

ENTER NAME/ID...

DATABASE HOME PAGE


HOME - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://localhost:8084/WebApplication1/home.html
GUI DESIGN FOR GENOME DATA COMPRE
ABOUT
ONLINE RESEARCH
ADMINISTRATOR LOGIN
CONTACTS
DECODE
WELCOME
http://localhost:8084/WebApplication1/online.html
Local intranet

NAME : p101

ID : potato

THE SEARCHED SEQUENCE IS GIVEN BELOW:

c1

SEARCHED SEQUENCE PAGE

IV. Hardware and Software Specification

Hardware Specification:
Server Configuration-

Computer Processor : Pentium IV
Hard Disk : 80GB
RAM : 1GB
Ethernet Card : 160 Mbps Intel LAN Card

Client System Configuration-

Computer Processor : Pentium II
Hard Disk : 4 GB free disk space.
RAM : 128 MB Ethernet Card : Any

Software Specifications:

Operating system : Windows XP.
Software : Turbo C

V. Experimental Results

Phases of the project

There are two phases involved in this whole project namely:

- COMPRESSING THE DATA
- DECOMPRESSING THE DATA

The whole task is being carried out by using TURBO C

A. *Compression phase*

- During compressing the data we are taking 4 characters at a time from the input file.
- The input file consists of combination of :
  - ❖ Adenine(a)
  - ❖ Thymine(t)
  - ❖ Guanine(g)
  - ❖ Cytosine(c)
- This four character are replaced by a single ASCII character

B. *Decompression phases*

- This phase involves the transformation of compressed data into its original format using the compressed file as the source.
- It involves reading the ASCII characters from the input file and converting them to its equivalent 'a','t','g','c' sequence iterating the loop four times.

Compression experiment

Take a source file which is to be compressed. For example " atgcattttttaaaaagcttgcacgtacttt " is the sequence present in a source file.

Run the program for compression technique. The program will asked for the source file name to be compressed , give the above file name. Again the program will ask for the one more file, over there it will store the compressed file converted into ASCII character.

By following the above step we compressed file as
"□U□□" i.e 75% size is reduced from the original form.

Decompression technique

Take the compressed file to be decompressed into the original form. "□ U □ □" is to be decompressed to original form, present in the compressed file.

Run the program for decompression. The program will asked for the file to be decompressed. Give the file name. Again it will ask for the file name to keep the decompressed data. After that the whole file is decompressed and The original form of data is stored in the file name given.
After the decompression we get the data in original form.
i.e " atgcattttttaaaaagcttgcacgtacttt ".

if we compare the original data with the data what we got is 100% loss less.

VI. Result Discussion

This algorithm runs almost 10^3 time faster, our algorithm performances better than it in both compression ratio and elapsed time. By just using the C technique, users can obtain original sequences in a time that can't be felt. Additionally, our algorithm can be easily implemented while some of them will take you more time to program.

VII. Conclusion

This compression algorithm gives a good model for compressing DNA sequences that reveals the true characteristics of DNA sequences. This method is able to detect more regularities in DNA sequences, such as mutation and crossover, and achieve the best compression results by using this observation.

VIII. Future Work

The whole concept can be implemented in any website and we can provide this concept online for user benefits, who work continuously with this GENOMES sequence, can easily send the sequence easily with less time and 100% loss less guarantee.

Also aim is to provide encoded data along with signature file at runtime that we can ease of the amount of usage of database.

Acknowledgement

The authors are grateful to all our colleagues for their interest and constructive criticism of this study.

References

[1] S. Grumbach and F. Tahi, "A new challenge for compression algorithms: Genetic sequences," J. Inform. Process. Manage., vol. 30, no. 6, pp. 875-866, 1994.

[2] M. Li and P. Vitányi, An Introduction to Kolmogorov Complexity and Its Applications, 2nd ed. New York: Springer-Verlag, 1997.

[3] R. Curnow and T. Kirkwood, “Statistical analysis of deoxyribonucleic acid sequence data-a review,” J. Royal Statistical Soc., vol. 152, pp. 199-220, 1989.

[4] S. Grumbach and F. Tahi, “A new challenge for compression algorithms: Genetic sequences,” J. Inform. Process. Manage., vol. 30, no. 6, pp. 875-866, 1994.

[5] É. Rivals, O. Delgrange, J.P. Delahaye, M.Dauchet, M.O. Delorme et al., “Detection of significant patterns by compression algorithms: the case of Approximate Tandem Repeats inDNAsequences,” CABIOS, vol. 13, no. 2, pp. 131-136,1997.

[6] K. Lanctot, M. Li, and E.H. Yang, “Estimating DNA sequence entropy,”in Proc. SODA 2000, to be published.

[7] D. Loewenstern and P. Yianilos, “Significantly lower entropy estimates for natural DNA sequences,” J. Comput. Biol., to be published (Preliminary version appeared in a DIMACS workshop, 1996.)

[8] T.Matsumoto,K.Sadakame and H.Imani, ”Biological sequence compression algorithm”, Genome Informatics 11:43-52 (2000).

[9] X. Chen, M. Li, B. Ma, and J. Tromp, “Dnacompress:fast and effective dna sequence compression,” Bioinformatics, vol. 18,2002.

[10] National Center for Biotechnology Information,http://www.ncbi.nlm.nih.gov

[11] Book : D. Adam, “ Elements of Data Compression”, published by Vikas Publishing House.