Forward Error Correction For DNA Data Storage

(Received May 5, 2018)

Team members:
Shubhangi Mehrotra- 201501124
Sajal Tiwari- 201501150

Abstract
As the storage of data in DNA has been a topic for research lately, we have implemented the idea proposed in the paper 'Forward Error Correction For DNA Data Storage' by Meinolf Blawat in 2016 during 'The International Conference on Computational Science'. We have encoded a file in DNA format at the sender's end, dealt with the errors in the channel and have decoded it the receiver's end exactly the same as encoded. We have used object oriented programming in Java and have made 13 classes for the implementation part.

1. Implementation

The program take and input file path and an output file path as arguments. The program then converts the media file to byte array using the mediaToByte class. Then this byte converted file is further converted to binary using the class byteToBinary. These binary representations are taken in the chunks of 8 and are sent for modulation in the class 'Modulation' where they are converted into DNA strings as per the algorithm specified in the research paper which is referenced and the same has been shown in Figure 1 and Figure 2 shown on the next page.

The two condition that had to be taken into account while converting binary into DNA strings were as follows:

i) The first three nucleotides shall not be the same.

ii) The last two nucleotides shall not be the same.

Therefore for each binary chunk of 8, we get a minimum of two valid strings. We send both of these strings into different clusters A and B so that we can refer to other in case of error. We do this for all the chunks and send them to clusters A and B in the format ABAB. ClusterToFile class is used to write these clusters into file which can be later used for error detection and correction.

We have introduced errors randomly in the DNA strings using RandomErrors class as might happen in the channel. These random errors are generated which make strings either invalid or wrong. The errors introduced are of the type insertion, deletion and substitution. We use an algorithm called editDistance, a dynamic programming approach which returns the number of operations needed to make erroneous string same as original one.

The error free code is sent to the demodulation class which takes the DNA strings and does a reverse mapping of the algorithm that was used while modulating the binary sequence as specified in Figure1 and Figure 2 on the next page. The resulted binary sequence is then converted into byte code in the class binaryToByte taking into account 8 binary bits which is in turn converted to a media file using the class byteToMedia and the file is regenerated at the source location as specified by the user.

Table 1: Mapping of first 3 bit tuple to 3 nucleotides

Value	Nucleotide
00	A
01	C
10	G
11	T

Table 2: Mapping of last 2 nucleotides to a pair of nucleotides

Value	Option1	Option2	Option3	Option4
00	AA	CC	GG	TT
01	AC	CG	GT	TA
10	AG	CT	GA	TC
11	AT	CA	GC	TG

2. Experimental Results

After certain experiments and on randomly introducing different kind of errors, we observed that the maximum file size that could be converted was 15 MB while the average number of error introduced on a 3 MB file were 1/3 of the total file size in which insertion errors were 0.27, deletion errors were 0.32 while substitution errors were 0.41.