DNA as a Storage Medium

Golam Md Mortuza
Department of Computer Science
Boise State University
Boise, Idaho, USA
golammdmortuza@u.boisestate.edu

Abstract:

Advances in information and communication technology are creating an explosion of data. Data storage architectures with the ability to store significant amounts of data over an extensive period of time are a critical need. High data density, longer information retention, and lower energy consumption make DNA a promising data storage architecture. However, because of the imperfection of DNA synthesis and sequencing, it is more error-prone than silicon-based storage systems. Providing a better error-correction algorithm and an efficient encoding/decoding scheme can make DNA a viable data storage system. This paper addresses the various issues regarding DNA based storage systems and discusses how these issues have been addressed and improved upon over time. Finally, an artifact is developed where we proposed a custom error-correction scheme for DNA.

Keywords**:** Nucleic acid memory, Error correction, Data storage, DNA origami

1.Introduction:

Technological advancement in the sector of electronic devices makes humans more potent than ever. As the prices of these electronic devices are reducing, people are using these devices more. The enormous propagation and usage of these devices are generating data at an exponentially increasing rate [1]. Every human will generate on average 1.7 megabytes of data per second, which will create 418 zettabytes of data in a single year. Furthermore, every year, the data generation rate is increasing by 50%[28]. This explosion of data is creating a storage crisis. Research shows that humans will run out of silicon supply by the year 2040 to store the projected generated data (3 X 10_{24} bits) [2].

Some types of data, like financial, historical, and CCTV footage, need to be stored for long periods but are not frequently accessed. However, current data storage systems are not reliable for the long term (> 30) storage, requiring continuous copying of this archival data to maintain its integrity. This, in part, has led to estimates that U.S. data centers will consume 138 billion kilowatt-hours of energy in 2020, requiring 50 mega coal-based power plants costing ~13 billion USD. The production of this energy will create 150 million metric tons of carbon pollution per year [3]. These

facts are driving the search for an information storage medium that can store data without replication for hundreds or thousands of years with very little energy consumption.

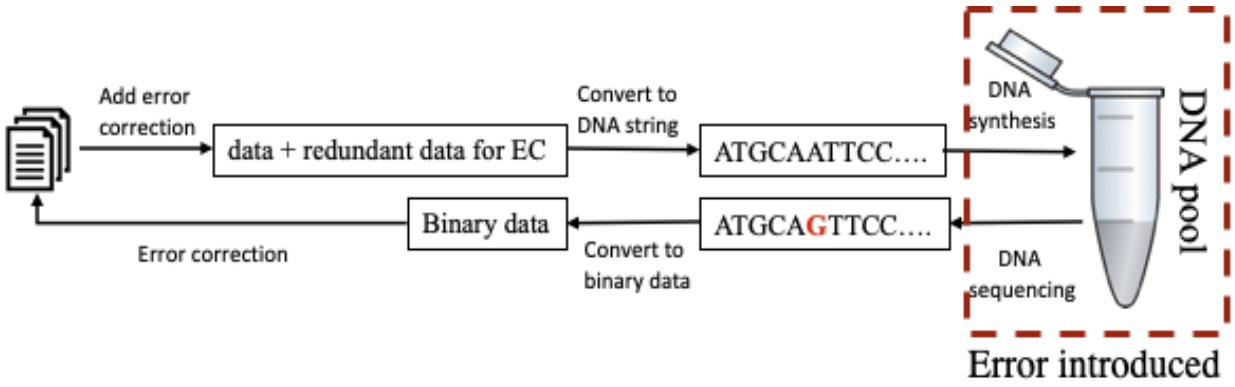

Figure 1: DNA storage process overview

The limitations of silicon-based storage are motivating researchers to explore other viable storage media. Deoxyribonucleic acid(DNA) is a promising medium of data storage. A single molecule of DNA is called a nucleotide. A nucleotide is one of four types of nitrogen bases: adenine (A), thymine (T), guanine (G), and cytosine (C). The digital data is converted into DNA sequences using a mapping scheme which consists of nucleotides. An error correction algorithm is introduced in this conversion process. During the decoding process, a decoder converts back the sequences into digital data. The decoder tries to fix any errors if found. An overall process is shown in figure 1. The storage density and retention rate of DNA are much higher, and the energy consumption is much lower than the silicon-based storage medium. The theoretical information density limit of DNA storage is more than 1 ExaByte/mm^3[10]. The volumetric density of DNA is 10^3 times more, and the energy consumption is 10^8 times less than flash memory. To store all the data available today, about 10^{22} bits, at the densities that DNA enables we need only a box of size 10 * 10 * 10 cm^3, and to store all projected data that will be generated until the year 2040 we need only ~1 kg of DNA[2].

These estimates assume ideal conditions and without the need for any error correction or duplication. However, in practice, we need to incorporate some error correction code and provide some duplication to make the storage system more robust. These will add some overhead to the DNA storage system. The amount of the overhead added will largely depend on the robustness and efficiency of the error correction algorithms, driving the development of space-efficient error correction schemes for DNA storage. However, even at the current state of error correction algorithms, the volumetric density of DNA based storage systems is much higher than conventional storage systems.

Further, the retention time of existing storage systems is around ten years. In the right conditions, DNA can be stable for more than 1000 years. The retention rate of DNA is $2*10^4$ years in water and $2 * 10^7$ years in the air at 10 degrees Celsius [2], which significantly exceeds the best retention

rates of current storage systems. Further, DNA can withstand a much broader range of temperatures (-800˚ C to 800˚ C) [31].

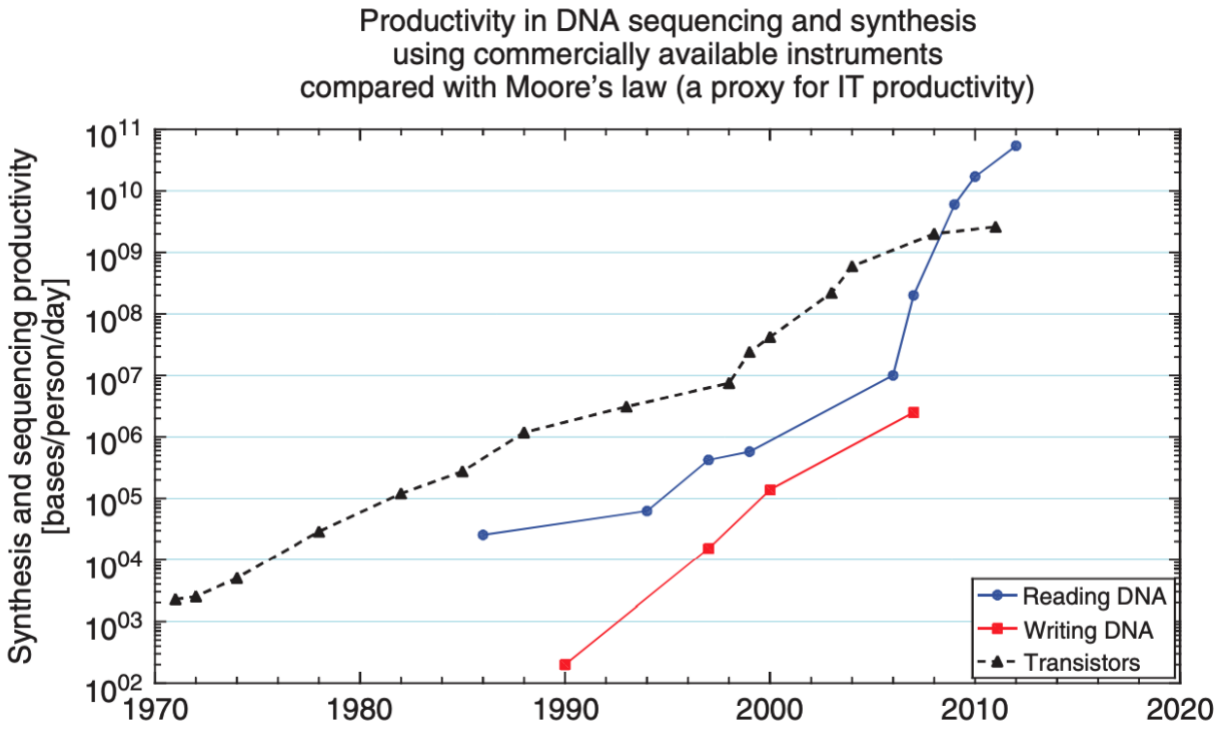

Figure 2: The graph shows the maximum productivity achieved for DNA synthesis and sequencing over time. Productivity is measured by the number of bases read by a person in a single day. Dotted lines indicate Moore's law [6] of the number of transistor changes in a chip per year.

Currently, storing data into DNA is considered impractical because of higher synthesis and sequencing costs. Nevertheless, both the synthesis and sequencing cost of DNA are decreasing faster than Moore's Law, and at current rates, the cost reduction trend will meet the needs of practical DNA storage shortly [29,30]. The current cost to read and write information into DNA is ~10^{-7} and ~10^{-4} USD per bit, respectively [2]. Along with the price, both synthesis and sequencing speeds are improving faster than Moore's law. Figure 2 shows the improvement of DNA sequencing and synthesis speed over time. Developing an efficient algorithm that can convert the digital data into the DNA bases and then read back the data from the DNA bases to digital data, could lead to a new era of data storage.

The idea of storing digital information in DNA was first introduced in 1988[27]. Where the author converted the pixel information of the image "Microvenus" into a 5 X 7 matrix of values 0 and 1. The converted information then encoded into DNA sequences and inserted into *E.coli*. The original image was successfully recovered from the DNA sequences. In 1999 researchers successfully encoded and recovered 23 characters [5]. These early attempts stored only a small amount of data.

In 2012 church *et al.* [7] first stored data in DNA on a larger scale. They successfully encoded and recovered 643 KB of data [7] in DNA. In the following year Goldman *et al.* stored even more data(739KB) in DNA [4]. Since then, the progress of DNA storage has been rapid.

1.1 Thesis Statement:

As storing data into the DNA is a relatively new concept, it has some limitations compared to the conventional scoring system. We will address some of the major limitations of DNA based storage systems in this paper.

Error Correction: DNA is a chemically active material with molecular reaction. Also, some patterns in the DNA strand, *i.e.*, repeating sequence, unequal GC content, palindromic sequence. can make both sequencing and synthesis more difficult. These issues also introduce errors in the DNA strand, which need to be corrected in the decoding steps. For that, we need a reliable error correction algorithm.

Cost: The cost of storing data is probably the most vital issue of a DNA based storage system. Though the cost is reducing faster than Moore's law, it is still much higher than conventional storage systems and is the main hindrance of conducting more research on this sector.

Rewrite ability and Random Access: Both rewrite ability and random access are important parts of any storing system. Though significant research is being conducted on these capabilities for DNA storage systems, it is still not possible to achieve these capabilities effectively.

2. The landscape of the storage system

In this section, we will discuss the limitations, advantages, and future of the current storage system. Several existing storage systems could be considered for long-term archival storage of information, but all of them have significant problems in the area of longevity and energy consumption. We will discuss these issues as well as the potentiality of DNA as a storage medium.

2.1 Advantages and disadvantages of the current storage medium

Magnetic tape is one of the oldest forms of data storage systems. Where data is stored in a long thin plastic film with magnetic coding in a sequential manner. So accessing the data is also sequential though random access is possible by rewinding the tape. Nevertheless, this process is time-consuming. Because of the random access issue, this medium is preferred for an archival storage system. However, the retention time is only 30 years, which is not enough for archival storage purposes. Low cost per bit makes it a viable storage system**.** European Organization for Nuclear Research stored 0.08 exabytes of data, which grew 15 petabytes every year. Ten percent of the data was stored in hard disk and remaining in magnetic tape. After 20 years, data started deteriorating. Thus they lost important data for the lack of having a better archival system.

Magnetic disks store data on a rotating disk where data can be read/written by a magnetic head. Random access is much faster than magnetic tape. The speed of the random access depends on the disk rotation speed. Which almost achieved the maximum theoretical limit. So further improvement on the speed for the magnetic disk speed is almost impossible. The retention time is also ~10 years. Which increases the maintenance cost of the data. Also, magnetic drives are error-prone to high temperatures, moisture, and exposure to magnetic fields.

Optical devices store the data in flat round discs where binary information is encoded by changing the photo-physical forms on the surface. Laser light or electromagnetic waves are used to read/write the data from/to the disk. The light/wave needs to be at the correct location to read/write the data. Because of that, optical devices also face the rotational limitation similar to the magnetic disks. These mediums are hardly used in data storage for a larger scale because of low data density. However, it is still quite popular for smaller-scale data storage for lower manufacturing costs.

Flash memory is getting popular day by day because of its lower read/write time. Integrated circuit assemblies are used to store data. As it has no moving parts, it has no mechanical limitation. Which makes it faster than other storing devices. However, the cost per bit for this medium, while coming down, is much higher than any other device described above. Also, the information starts degrading if it is not fed power for more than a few months.

2.2 Effects and future of current storage system

The collapse of Moore's law:

Gordon Moore stated that the number of transistors in a chip would double every two years on average. This statement is also known as Moore's law. Rapid advantages of technology proved the validity of Moore's law. In April 2005, Gordon Moore himself stated that the projection could not be sustained indefinitely. Ultimately humans will reach the minimum level of transistor size. In 2012 Michio Kaku predicted the failure of Moore's law within the next decade. So humans should search for alternative materials and approaches for computing as well as storage.

Silicon pollution:

As the production of silicon-based products increases, the pollution-related to those productions also increases. Semiconductor industries use a large number of toxic chemicals to manufacture their components, including a disk drive, circuit board. A statistic published by the U.S. Department of Labor's Bureau stated that workers of semiconductor industries have a rate of occupational illness twice as high as the workers of other manufacturing sectors. The water consumption of the chip manufacturing industry is also high. Moreover, all the data storage mediums have limited lifespans. So the damaged storage medium needs to be dumped. The dumping process can also create environmental pollution. Production of energy that is required by these storing medium causes lots of environmental garbage.

2.3 DNA as a storage medium

Limitations of the current storage medium motivate the search for a newer form of data storage. It seems like nature has already developed a solution for this purpose. DNA has an extraordinary density for storing data. A single gram of DNA can contain 108 terabytes of binary data. Most of the storage mediums store data in a linear or planar manner while DNA stores the data in a volumetric fashion, which makes it denser. DNA consumes much less energy (10^{-10} W/GB) compared to the other storage mediums. Further, while other mediums start losing their integrity within a few years, DNA can maintain its integrity for at least a few hundreds of years. All these benefits of DNA make it an attractive medium of storage. Table 1 shows a comparison between DNA and other storage mediums.

Memory	Retention(Years)	On power(W/GB)	A.Density(bit/ cm^2)	V.Density(bit /cm^3)	Latency(μs/bits)
Flash	~10	~0.01-0.04	~10^{10}	~10^{16}	~100
Hard Disk	> 10	~0.04	~10^{11}	~10^{13}	~3000-5000
Magnetic Tape	~30	~.004	~10^{9}-10^{10}	~10^{12}	~60-200
Cellular DNA	~100	< 10^{-10}	unavailable	~10^{19}	<100

Table 1: Comparison between cellular DNA and conventional Data storage system

3. Issues with DNA based storage system

In this section, we will discuss each of the major issues of DNA and current research that is being conducted to solve those major issues.

3.1 Error Correction

The error rate of DNA is higher than other conventional storage systems. Errors occur due to various factors like DNA synthesis imperfections, PCR dropouts, degradation of DNA molecules over time, stutter noise, and sequencing errors [9]. Most of the errors occur in the DNA because of the structure of the DNA strand. Oligos with more than 60% GC content exhibits higher dropout rates that lead to PCR errors [10]. Strands that have ~50% GC content show more stability than unequal GC content strand [13]. Insertion and deletion errors start arising if the homopolymer run of a strand is more than 4 nt [11]. Also, because of homopolymer PCR, slippage errors are noticed

[12]. The effects of errors for GC content and homopolymers are shown in figure 3. The presence of a single long sequence in multiple strands can create a hairpin structure, which makes the sequencing more difficult and error-prone. To avoid hairpin structure creation, both the repeating sequences and palindromic sequences need to be avoided. Oligos which do not have any of the aforementioned structural problems show a lower rate of synthesis and sequencing errors, but still, have higher error rates than competing memory storage technologies [1]. In order to fix these errors, an error correction algorithm must be incorporated in the encoding and decoding process, adding some redundant data in each oligo that will ensure the integrity of that strand. DNA storage can be made more robust by using a better/robust error correction algorithm.

Fortunately, several useful, applicable error correction algorithms already exist, which are heavily used in noisy channel communication as well as in the conventional data storage system. Hamming code, Reed Solomon code, and LDPC code are the most famous error correction algorithms. One common problem with all these error correction algorithms is handling insertion and deletion error. While these error correction algorithms work great for handling mutation errors, because of the frameshift, these error correction algorithms can not fix the insertion and deletion error.

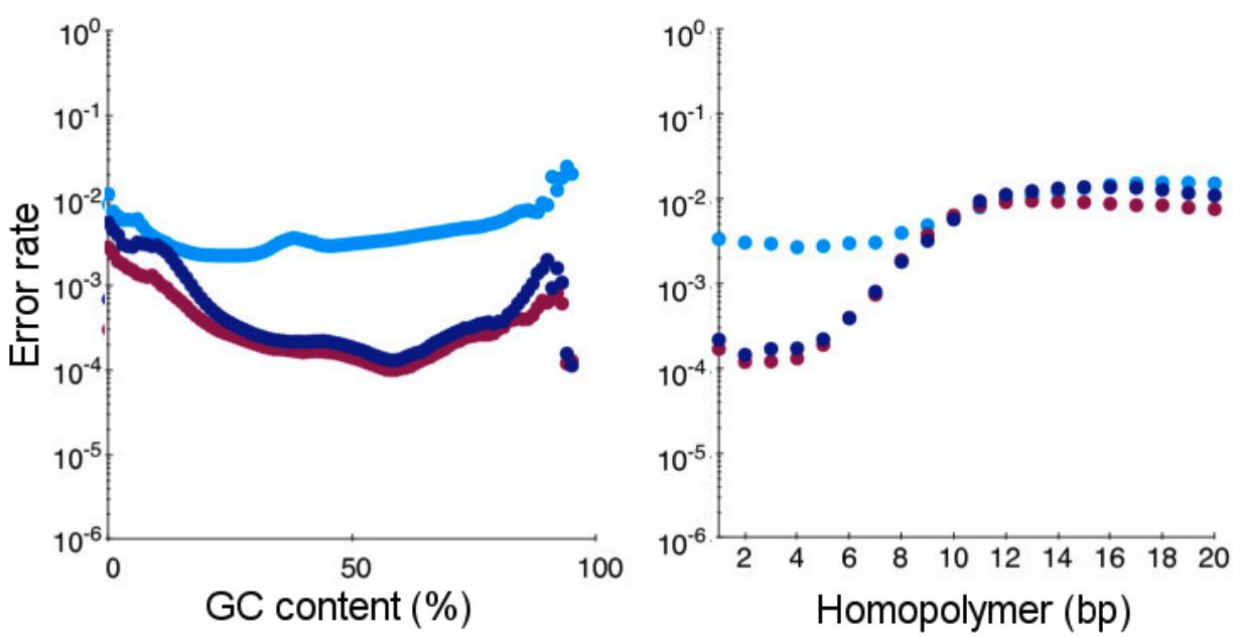

Figure 3: The effect of GC content and homopolymer length on error rates for Illumina sequencing, where blue indicates mutation error, dark blue deletions error, and purple insertion error.

Hamming code [19] is a member of the linear block code family, which was developed in 1950 by Richard W. Hamming. Hamming codes can detect at most two bits of error in the data block and can fix one bit of error in the data block. Because of these limitations, hamming codes have hardly been used in DNA based storage systems. A generator matrix *G* of shape *n x k* is created to encode the data using a hamming code. Where *n = code block size and k = number of redundant bits*.

$$G := (I_k \mid -A^T)$$

Where I_k is the identity matrix of the shape *k*, matrix A indicates the linear relationship between data bits and the parity bits. Transmitted message vector $\underline{x} = (\underline{a}G) \ mod \ 2$. Where $\underline{a}$ is the original message vector. H is the parity check matrix. Which is also called a hamming matrix. This matrix is used during the decoding of the message.

$$\boldsymbol{H} := (A \mid I_{n-k})$$

By definition $(HG^T) \ mod \ 2 = 0$. Let *r* be the received message vector. *H* is multiplied with $\underline{r}$ to get the syndrome vector. The syndrome vector indicates the position of the error bits if there are any. By altering that position, the correct message can be reconstructed. If there is no error in the transmission message, then $(H\underline{r}) \ mod \ 2 = 0$(zero vector).

A 5 bytes message "HELLO" was stored and retrieved successfully by Takahashi *et al.*[24] in 2019. Hamming code[19] was used as an error correction scheme for this system. To check the integrity of the message, the last 12 bit of the SHA256 hash of the original message "HELLO" was appended to the original data. Then the new data was passed through a one time pad to increase the entropy of the data. The data were then coded using a two-layer scheme that stored 5 bytes over 32 dsDNA bases. The outer layer consists of (31, 26) Hamming code, which can detect two base-read errors and corrects all single-base errors. They used modulo four arithmetic, and the following data-nucleotide mapping was used. A = 0, C = 1, G = 2, T = 3.

Errors like deletion and read truncations made it difficult to retrieve the original data. Out of 25,592 reads, 286 reads were aligned well and contained enough bases to attempt decoding. Of those 251 were uncorrectable, 11 had invalid hashes after correction, eight were corrupted but correctable, and of those three had hashed in agreement, 16 were perfect reads, and 0 were decoded but contained wrong information.

Repetition codes are the most basic form of error correction scheme, which is also known as Multiple sequence alignment in DNA applications, where a single data block is repeated a predetermined number of times. Examples of repetition code are shown below, where a single message is repeated four times:

$$[X] \mapsto \begin{bmatrix} X \\ X \\ X \\ X \end{bmatrix}$$

If one or more (a minority) data blocks differ with another, then that particular data block is considered as corrupted. Majority voting is used to select the correct data block. Though this scheme is simple, it is not efficient as the amount of overhead is very high. Also, if the same kind of error affects all data blocks, then it is not possible to detect that error. Despite this scheme inefficiency, researchers have used it to store data into DNA. Goldman *et al.* [4] and Bornholt *et al.*[1] used this scheme to correct their errors.

In 2013 Goldman *at. el.*[4] stored 0.75 megabytes of data into DNA. They used repetition as an error correction algorithm. They also used a limited amount of error detection using a checksum,

which is also called parity trait. The parity trait was used in each strand to check the integrity of that strand. However, because of not having a better error correction algorithm, they lost two sequences of 25 nucleotides. So in the decoding process, manual intervention was required to recover the file. Because of the overhead of the repetition code, they only achieved information density 0.33 bits/nt and physical density 2.25 petabytes/gram.
Bornholt *et al.* [1] stored 0.15 megabytes of data into DNA. They utilize Goldman's encoding and decoding algorithm. They also introduced an XOR method of encoding where the redundancy level of any portion of the data can be controlled. Mutation errors can be detected with this method but impossible to correct as it is not possible to determine which strand the error is actually in. Because of the weak error correction scheme, they also had to manually intervene in the decoding process to recover the entire file. For higher overhead of repetition, their information density was also 0.88 bits/nt, where the coding potential was 1.58 bits/nt.

Reed Solomon code(RS code)[21] is a special class of BCH[20] code, which was first introduced in 1960 by Irving S. Reed and Gustave Solomon. Since then, it is the most popular and widely used error correction algorithm. RS codes are capable of correcting both burst errors and erasures. It can detect/correct multiple errors. RS codes are used in current conventional data storage systems like CD, DVD, and hard drive. It is also widely used to communicate over the noisy channel [14]. RS code can both detect and correct bit flips, which is also known as mutation error at arbitrary locations. However, it can not handle insertion and deletion errors because of the frameshift problem. If the message size is *n* bytes, RS code adds a redundant data of size *k* bytes at the end of the message. RS code can detect *k* bytes of errors at arbitrary locations and can correct up to $\lfloor k/2 \rfloor$ bytes of errors at arbitrary locations. One of the advantages of RS code over the other error correction scheme is the length of redundant code block *k* can be changed depending on the usage or by analyzing prior error patterns. For example, errors frequently occur in the noisy channel, so the value of *k* should be higher if we apply RS code in a noisy channel communication. Several researchers have used the RS code to detect/correct errors in DNA based storage systems[25, 14, 9, 26].
A generator matrix ***G*** is created to encode the data using the RS code. The size of the generator matrix is *n*+*k*-*by*-n and it must have two fundamental properties (i) first *n* rows constitute an *n* x *n* identity matrix (ii) any *n* of the *n* + *k* rows are linearly invertible, so they are invertible. Property (ii) can be achieved by using the Vandermonde matrix, which has the following form:

$$\begin{bmatrix} 0^0 & 0^1 & 0^2 & \dots & 0^{n-1} \\ 1^0 & 1^1 & 1^2 & \dots & 1^{n-1} \\ 2^0 & 2^1 & 2^2 & \dots & 2^{n-1} \\ \dots & \dots & \dots & \dots & \dots \\ (2^{n+k}-1)^0 & (2^{n+k}-1)^1 & (2^{n+k}-1)^2 & \dots & (2^{n+k}-1)^{n-1} \end{bmatrix}$$

Property (i) can be achieved by performing a series of linear transformations over the Vandermonde matrix. An example of a generator matrix is:

$$G = \left[\frac{I_n}{D}\right]$$

$$G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 6 \\ 4 & 3 & 2 \\ 5 & 2 & 2 \end{bmatrix}$$

The data is encoded by matrix multiplication operation of generator matrix G and the column vector of the data $\underline{a}$. So the encoded message $\underline{x} = G\,\underline{a}$. During decoding the matrix, the inverse of the generator matrix is used. However, no inverse operation is applied in the identity portion of the generator matrix. So the shape of the inverse generator matrix is:

$$G'^{-1} = \left[\frac{I_n}{D^{-1}}\right]$$

$$G'^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 5 & 6 & 2 \\ 5 & 7 & 3 \\ 3 & 3 & 3 \end{bmatrix}$$

The received data vector($\underline{r}$) is multiplied with the inverse generator matrix to get the original data back. So the recovered matrix $x' = \underline{r}G'^{-1}$.

Grass *et al.* [25] used two levels of RS code to detect/correct errors. Figure 4 describes their encoding method. The inner RS code corrected an average of 0.7 nt errors per DNA strand(158 nts). The outer RS code had to account for a loss of 0.4% of total sequences and corrected about 0.4% of the sequences. The outer code can handle the loss of about 17% of all complete segments if they still contain errors after the inner RS correction code is applied. That provides an error-free file recovery without any manual intervention. They successfully stored and recovered 83 kilobytes of file. Because of using RS error correction, they were able to get rid of the repetition

code, which increased their net information density over Goldman and Bornholt. Their net information density was 1.14 bits/nt.

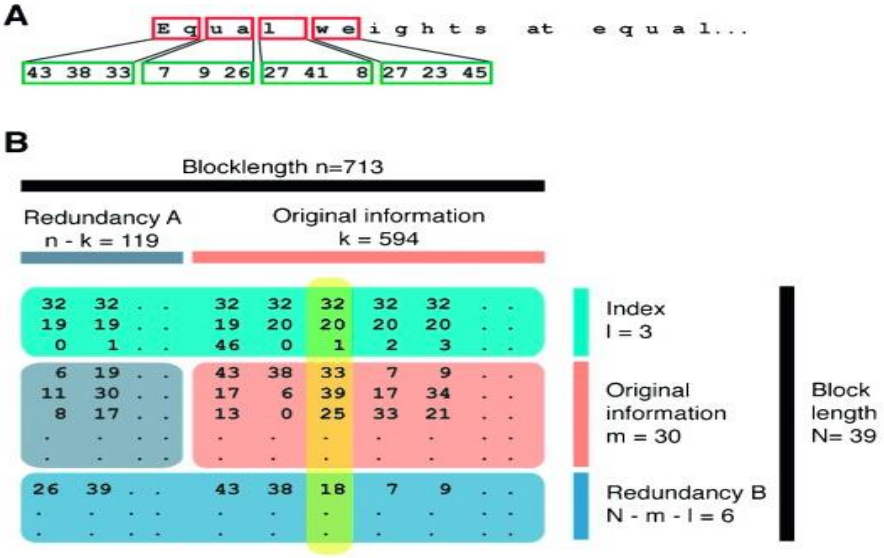

Figure 4: (A) By base conversion, two letters of the text file are converted into three numbers of Galois Field of size 47 (GF(47)). (B) The index (bright turquoise) is also converted into a number of GF(47). At first, the outer level of the RS code(glacier) is applied only in the data block (rosebud). Then the inner level of RS code(malibu) is applied over the index, data, outer level RS. At the last step, this whole matrix was converted into DNA bases. Each column(yellow) of this is an individual DNA sequence.

Blawat *et. al.*[14] incorporated three levels of error correction codes. i) Bose-Chaudhuri-Hocquenghem(BCH) for address sections ii)a 16-bit Cyclic Redundancy Code(CRC) for data sections iii) A RS for the entire data block. Their decoding process has four different stages. In the first stage, the oligos were selected based on their correct length, the CRC checking succeeded and the BCH code did not show any error for addressing and the parity bit was correct. In the second stage, remaining erroneous oligos were examined by majority voting. At first majority, voting was done with the oligos that have the correct length. If the length of any oligo differs by one, then a brute force method was used to find the possible insertion/deletion of that oligo. At the third stage, the decoder attempts to reconstruct correct oligos from the remaining oligos whose length differs by more than one. This was done by combining all oligos with the same address, which could not be decoded successfully in the first two passes. At the last stage, there was a possibility that some of the oligos were still missing. So RS was used to recover the correct oligos and correct the wrongly corrected oligos. RS code was able to decode all the data blocks in all libraries successfully. While robust, because of using a high amount of error correction, the Blawat algorithm has less information density(0.92bits/nt) than the Grass algorithm(1.14 bits/nt).

Low Density Parity Check(LDPC) code is a class of linear error correcting code, which was developed by Robert G. Gallagher in 1962, but it started gaining popularity for the last ten years.

Because of the ability to decode hard bits and soft bits, it is being used in lots of sectors nowadays. Soft bit decoding provides a probability of a bit being 0 or 1 where hard bit decoding provides if a bit is 0 or 1. This code can be characterized by the sparsity of the parity check matrix(LDPC matrix), which provides the ability of having a large minimum distance of the code hence improved performance.

Let a message vector $a = [\, d_1\ d_2\ d_3\ d_4\ d_5\ d_6]$ and the transmission vector$x = [\, d_1\ d_2\ d_3\ d_4\ d_5\ d_6\ p_1\ p_2\ p_3\ p_4\ p_5\,]$. The parity*(p1,p2,.....)* of the transmission vector is calculated by the parity check matrix *H*. This matrix contains mostly 0's and only a small number of 1's. That is said the matrix is sparse. This is where the name low density came from. Matrix *H* indicates that the parity bit p_1 has the XOR value of data bits d_1, d_2, and d_3.

$$\mathbf{H} = \left(\mathbf{C}^{\mathrm{T}} | \mathbf{I}\right) = \begin{array}{c} \begin{array}{cccccc|ccccc} d_1 & d_2 & d_3 & d_4 & d_5 & d_6 & p_1 & p_2 & p_3 & p_4 & p_5 \end{array} \\ \left(\begin{array}{cccccc|ccccc} 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{array}\right) \end{array}$$

By using transmission vector *x* LDPC code generated channel code. While doing this it follows some rules. (i) no distinction is made between parity and data bits (ii) at least one parity bit is involved in each parity equation. (iii) each parity bit must appear at least twice in the parity equation. Figure 5 shows an example of the channel code of the LDPC code. The variable bits (*x)* are the transmission bits (shown at the top row). Module 2 of the check nodes must be a 0 vector. The check nodes are not transmitted. However, the decoder of LDPC code can generate the same check nodes connection using the code parameters.

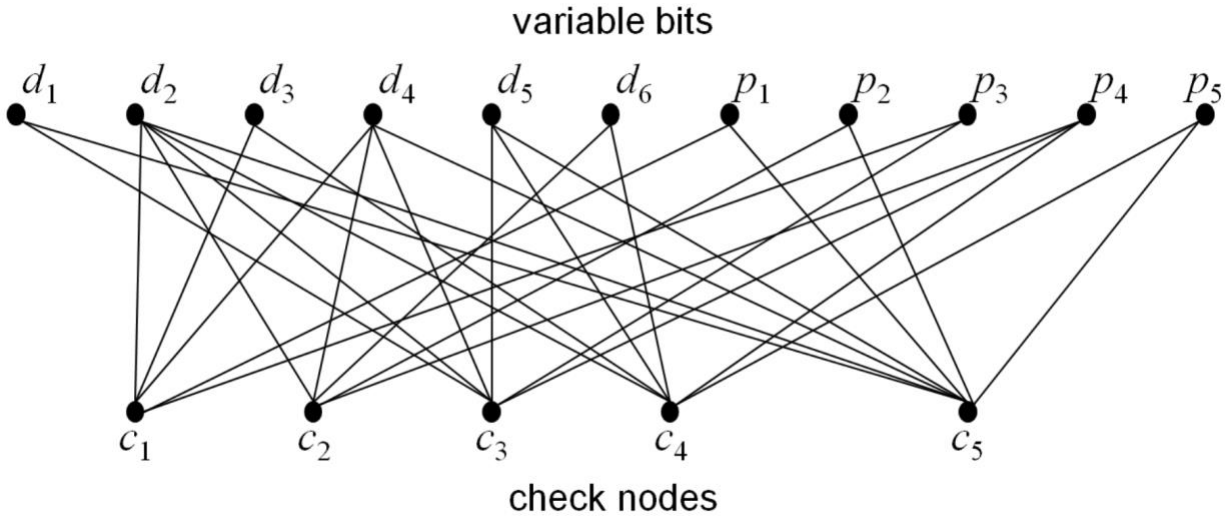

Figure 5: Graphical representation (tanner graph) of LDPC channel code.

Using the LDPC code Aldrin *et al.* successfully stored a picture of size 438 bytes. The authors also performed simulations to show the efficiency of the error correction capability of the LDPC code. Their simulation result is shown in figure 6. Researchers also proposed a different version of LDPC codes whose primary focus is to correct errors in the DNA storage system [35]. Shubham *et al.* [36] used three different error correction algorithms in their encoding/decoding scheme. At the first level, LDPC codes were used as the outer level error correction. The block size of each individual LDPC code was 32 kilobytes. The data along with the LDPC redundancy was converted into DNA bases from binary data. In order to handle insertion and deletion errors, they inserted a synchronization marker (sequence "ATG" was used for this purpose) in the middle of the data block sequence. At the final step, index bases were added to the sequence. These index bases used the BCH error correction algorithm. At the decoding step, BCH codes are used to decode the index first. *6k* bits of redundancy are required to recover *k* bits of errors by BCH codes. Multiple sequence alignment (MSA) was used to reduce the errors in any individual sequence. During this process, if a sequence does not have a correct length then a synchronization marker is used to recover the sequence. If the synchronization marker was shifted left by 1 base then only the right part of the marker was considered valid. However, synchronization markers would not provide any information if both insertion and deletion occurred on the same side in the same amount. At the very last step, LDPC codes are used to recover the whole sequence (32 kilobytes). The authors observed that the sequences were much more robust with higher redundancy (50%) of LDPC code than lower redundancy of LDPC codes(10%) which was expected. Furthermore, the synchronization marker improved the reading performance by 10% while it reduced the writing performance (density) by 2-3%.

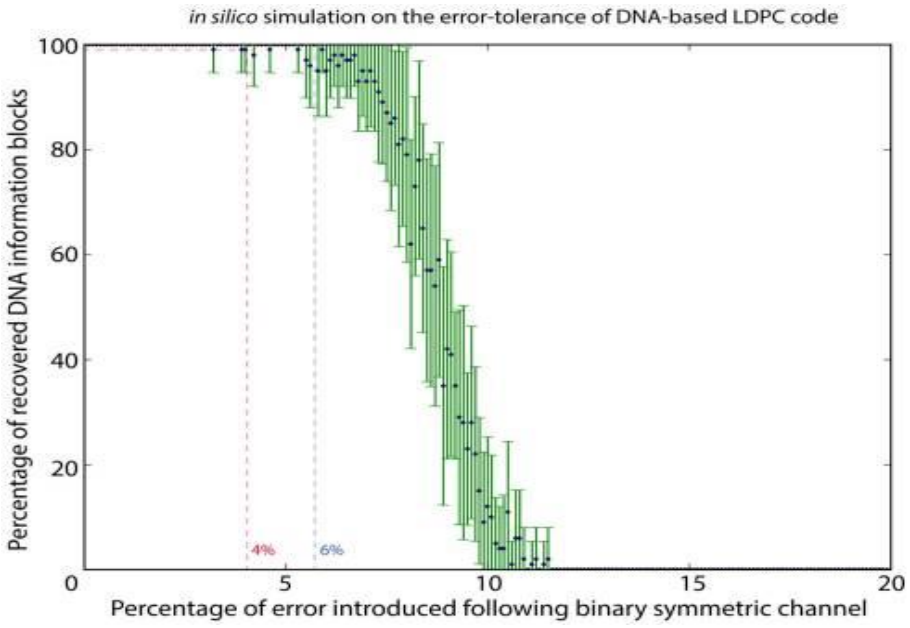

Figure 6: The efficiency of the LDPC code on DNA based storage systems.

As DNA has more error tendency than the regular data storage medium, error correction of DNA should be more resilient than other mediums. Not all kinds of error correction algorithms are not suitable for this purpose. We have only discussed the algorithms that researchers have used practically so far to store data. Table 2 shows a comparison between these algorithms.

	Hamming code	Multiple sequence alignment	Reed Solomon	BCH	LDPC
Error correction	Single bit	Multiple bits	Multiple bits	Multiple bits	Multiple bits
Error detection	Two bits	Multiple bits	Multiple bits	Multiple bits	Multiple bits
Error types	Scatter	Scatter + burst	Burst	Scattered	Scattered
Soft bit decoding	no	yes	no	no	yes
Minimum distance	3	no minimum distance	r + 1	2t + 1	no minimum distance
Application	Error-correcting code memory	because of low performance hardly used in any sector	QR codes, RAID 6, WiMax, DVDs, Blue rays etc.	Satellite communications, DVDs, SSDs etc.	Wi-fi, 5G, satellite transmission etc.

Table 2: Comparison of different error correction algorithms. For minimum distance for Reed Solomon, r = number of redundant bits used for Reed Solomon code. For minimum distance for LDPC code $t \geq \frac{Number\ of\ redundant\ bits}{2^m}$, $2^m = block\ length + 1$

Fountain code [9] is an information coding scheme that increases the robustness to dropouts. It is a class of erasure codes which is also known as rateless erasure codes. Fountain codes generate an infinite (potentially) number of smaller chunks of data, which are also known as a droplet. XORing segments of the data generate droplets. Fountain codes are typically used for transmitting data over a noisy channel. Original data can be reassembled from any subset of the droplets once

enough of them are correctly received. The name of the fountain comes from the water fountain from which glass is filled with water by catching water droplets from the fountain(i.e., encoded data can be recovered) once enough droplets are caught regardless of which droplets are in the glass or the order in which they were received.

Fountain code is not an error correction algorithm. However, it works better with dropouts. If we lose some arbitrary portion of the transmitted data, we will still be able to recover the whole file. Fountain code is often used along with other error correction algorithms, where the error correction algorithm verifies the integrity of any droplets. If the error correction algorithm cannot verify a droplet, then fountain codes throw out that droplet (do not use that droplet in the decoding process). Researchers have used fountain codes to store data in the DNA[9, 26].

Erlich *et al.*[9] first introduced fountain codes in DNA based storage systems. Fountain code was used to recover the whole file even if some of the sequences were missing. At the first stage, droplets were generated using the Soliton distribution to select segments for XORing. A unique seed and a RS block was attached with each droplet. Then the whole droplet was converted into a DNA sequence. At the time of decoding, RS code checks the integrity of each individual droplet. Though RS code can detect/correct errors, the authors decided only to use it to detect the error of each individual sequence. 72,000 numbers of oligos were generated to store 2.15 megabytes of file. Because of having a better error correction algorithm (RS) and handling the dropout (fountain code), this algorithm needed much less redundancy (7%) than other algorithms. So this encoding/decoding scheme had an information density of 1.57 bits/nts. And the physical density of 214 petabytes/gram. The combination of fountain code and RS code makes the whole decoding process more robust.

Due to the imperfection of DNA synthesis and sequencing, this storage medium is more error-prone than the conventional storage system. These errors still require a significant effort to correct. Over the last half-century, DNA synthesis has been improved continuously. [22] Important biotechnology applications like genomics and the development of smart drugs are expected to continue the development of DNA synthesis and sequencing, eventually making DNA a viable storage medium. However, with the current error rate, we can still store and recover data in DNA [9, 25, 14].

3.2 Cost:

Although a DNA storage system has multiple advantages over the conventional storage medium, costs, and times of writing and reading currently make it impractical to store data. However, the costs of DNA synthesis and sequencing are dropping at an exponential rate of 5- and 12-fold per year, which is much faster than electronic media at 1.5 fold per year.[8]. The cost of storing data into DNA depends on the efficiency of the encoding scheme. A better encoding/decoding scheme requires less amount of overhead, which reduces the cost. In 2013 Goldman *et al.* [4] required USD 12,400 to encode 1 megabyte of the file where their coding potential (encoding efficiency) was 0.88 bits/nts. But if they were able to achieve encoding efficiency of 0.94 bits/nts their cost would have reduced to USD 7,440/megabytes. In 2017 Erlich *et al.*[9] performed their wet-lab

experiment, which cost ~3500 $USD/megabytes. Which is almost ¼ th of Goldman's cost. Moreover, this improvement occurred only in four years. In 2019 Anavy *et al.*[26] used composite DNA in the synthesis cycle, which increased the logical density of the DNA and ultimately led to the synthesis cost reduction per megabyte.

Though the cost of DNA synthesis is much higher, the maintenance cost is significantly lower than conventional data storage systems. For example, if a data center stores 10^9 bits of data in a tape, it will require ~1 billion and hundreds of millions of kilowatts of electricity to build and maintain the data center. DNA can reduce the cost by three orders of magnitudes [33]. The cost of a DNA storage system can be reduced in two ways: first continuous improvement of DNA synthesis and second exploring quick-and-dirty oligo synthesis methods that consume less machine time and fewer reagents. Figure 7 shows the cost of DNA synthesis per megabytes. Clearly, the cost is reducing more than Moore's law.

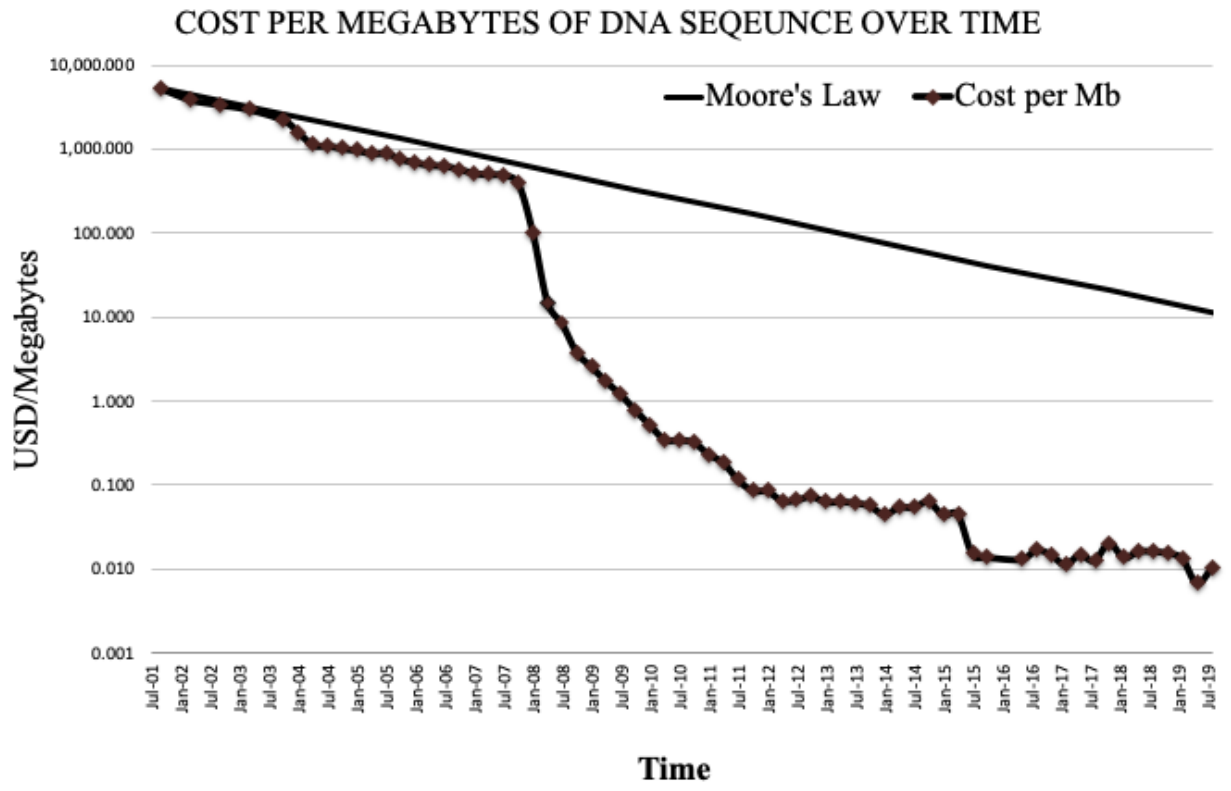

Figure 7: Cost of DNA synthesis per megabyte over time

The cost, efficiency, read/write time of storing data in DNA is directly proportional to the cost, efficiency and read speed/sequencing throughput of DNA sequencing, respectively. Based on the underlying mechanism, DNA sequencing can be divided into three generations of sequencing: Sanger sequencing, high-throughput sequencing/Next Generation Sequencing (NGS), and single-molecule sequencing.

The first DNA sequencing technology is called Sanger sequencing, which was developed by Frederick Sanger in 1977. It is considered as first-generation DNA sequencing or dideoxy/capillary electrophoresis sequencing. This method was very expensive, which cost around USD 2,400,000/billion bases. It also required a time-consuming method of plasmid cloning.
NGS was introduced in the last decade, which allowed us to perform sequencing on a large scale in parallel. It enables the simultaneous sequencing of thousands to millions of DNA molecules with short read lengths simultaneously. This system is capable of analyzing millions or even billions of sequences at the same time. NGS has reduced the cost of sequencing at a rate four times, as predicted by Moore's Law [28]. Using NGS, it is possible to sequence the entire human genome in a single day, where it will take over a decade to decipher the human genome by Sanger sequencing technology [32].
Single-molecule sequencing, also known as long-read sequencing, is considered a third-generation sequencing technology and has significantly increased the read length and raised the read speed. It works by reading the sequence at the single-molecule level. As this method is still in development mode, the cost and error rates are higher than the NGS technologies. However, both of them are improving over time. Table 3 shows a comparison between these three sequencing technologies.

Sequencing Technology	1st Generation	2nd Generation	3rd Generation
Cost (per kb)	1-2	10^{-5} - 10^{-3}	10^{-4} - 10^{-3}
Error rate	0.001 - 0.01%	0.1 - 1%	~10%
Sequencing length	1kb	25-150 bp	200 kb
Read Speed	~10^{-1}h	~10^{-7} - 10^{-4} h	~10^{-7} - 10^{-5} h
Sequencing throughput	1kb	10^{8} - 10^{12} bp	10^{9} - 10^{13} bp

Table 3: Comparison of three generations of DNA sequencing technology [33]

3.3 Rewrite ability and Random Access

Random-Access is one of the challenging issues in DNA based storage systems. In 2001 Kac *et al.* [34] first proposed an idea of information DNA(iDNA), which consists of information and single poly primer key(PPK), forward and reverse primer, and common 5-6 bases space to indicate stored information. PPK works as an address identifier. Encoding was done by mapping the ternary codes to only three bases A, C, and T while the sequencing primer consisted of all four bases with four positions as G to prevent mispriming. During the decoding process, PPK is decoded first to get the forward and backward primer; then information is retrieved. The author encoded 561 bits

of data. The main drawback of this method is that the decoding process needs prior knowledge of the primers.

In 2018 Organick *et al.*[37] implemented random access in DNA on a large scale. Where they designed a primer library for PCR-based random access methods using an in-silico process. The primer works as an address identifier in this process. They were able to retrieve 35 files(total > 200 MB) independently without error with fewer synthesis cycles. Their proposed methodologies can scale to physically isolated pools of several terabytes each. It is worth mentioning that this work attempted to store the highest amount of data in DNA molecules so far.

Yazdi *et al.* [13] reported encoding parts of Wikipedia pages of six universities in the USA. A total of 32,000 bps were needed to encode 17 KB of a file, which cost USD 4023. The estimated storage density is $4.9X10_{20}$ B/g for this scheme. A unique address design and encoding scheme enable the random access of the data. The address sequence is required to satisfy four constraints: (C1) Constant GC content (≈ 50%) (C2) Large mutual hamming distance (C3) Uncorrelatedness of the address (C4) absence of secondary structures. This address works as a unique identifier for each block they are associated with. Prefix-synchronized codes [17] have been used to avoid address patterns in the codewords. Prefix-synchronized coding also supports limited error-detection/ error-correction. Each of the addresses ends with the same nucleotide which is not used anywhere in the codewords, which helps to determine the ending of the address and start of the codewords. For correcting the error BRDS[16] error-correcting has been used.

Sharon *et al.*[15] introduced a novel technique to store dehydrated DNA spots on the glass cartridge. Liquid DNA does not provide the ability to have isolated DNA pools but dehydrated DNA provides that opportunity. Each of the pools can be accessed separately by digital microfluidics (DMF) which provide random access to that pool data. Digital microfluidics is a liquid handling technology that provides individual control over a droplet on an array of electrodes via electrowetting-on-dielectric phenomena. It is possible to use the same primer in different pools if the pool is isolated from each other, which provides more addressing capacity hence bigger file storage capacity. Each of the glass cartridges can store 50 TB of data. The cartridges can be further organized to provide a multidimensional addressing system. As each pool is different and isolated there is a possibility of contamination during accessing the pool. Oil, neighboring spot and path contamination are the contamination sources that were studied. A very little file contamination was found for oil and neighboring spot contamination. However, no proof of contamination was found along pathways.

Until 2019 all DNA storage systems needed human intervention for synthesis and sequencing. But If we want to use a DNA storage system on a commercial basis in the future it is not feasible to synthesize or sequence DNA every time manually. Takahashi *et al.* [24] introduced a fully automated DNA synthesis and sequencing technique for the first time in 2019, where authors successfully stored 5 bytes of data and read back that data automatically. This opens a new door for DNA storage systems. Their write to read latency is ~21 hours where the synthesis process consumed most of the time. Each base pair took ~305 seconds to synthesize. By adding more heat

in the cleave step the synthesis process could be reduced by 10-12 hours. Their system cost around 10K USD which could be reduced significantly by optimizing and calibrating the sensors.

4.Conclusion

Publish year	File size (MB)	Full recovery	Handle error	Data density (bits/nt)	cost(USD/MB)	Physical density(Pbytes/gram)	Main contribution
1988[27]	.000004	No	No	-	-	-	Introduced DNA as a storage medium
1999[34]	.00009	No	No	-	-	-	Store a sentence for the first time
2009[36]	.0002	-	Yes(Special form of Huffman coding)	.44	-	-	Stored multiple files (text, music, images), Unique primer design
August 2012[7]	.65	No	No	.83	-	1.28	Store data in DNA in larger scale
February 2013[4]	.75	No	Repetition	.33	12,400	2.25	Introduced data compression, Handle error by repetition
February2015[25]	.08	yes	RS	1.14	31,250	25	Retrieve the data without manual intervention, implement RS code for EC
September 2015[13]	.017	yes	BRDS	-	2,36,647	$4.9 \times 10_5$	Unique address scheme, Random access, rewrite ability
April 2016[1]	.15	No	Repetition	.88	-	-	Demonstrated random access for smaller scale
June 2016[14]	22	Yes	RS	.92	-	-	Error free retrieval of the data in a larger scale
March 2017[9]	2.14	Yes	Fountain + RS	1.57	3500	214	Incorporated fountain code in DNA, get highest data density
March 2017[37]	200.2	Yes	RS	1.1	-	-	Random access in larger scale
July 2017[38]	.0036	Yes	LDPC	1.72	-	1.1 X 10_8	Portability in DNA storage
March 2019[24]	.000005	yes	Hamming	-	$10000(equipment cost)	-	An automated device for synthesis and sequencing
September 2019[26]	21.4	Yes	Fountain + RS	1.93 - 4.29	-	2.92 - 5904.42	Introduced composite DNA letters

Table 4: Comparison of DNA storage coding scheme over time

DNA is still not a viable medium of storage in commercial uses yet. Some DNA sequence combinations – especially the repeated sequences, unequal GC content, palindrome sequences – are more error-prone than others. The presence of these sequences in the DNA makes it harder to recover. However, handling these issues is improving over time by either avoiding these sequences or fixing the error caused by these sequence combinations. We can see the improvement of this medium over time in table 4. The first approach in 1988 by Joe Davis introduced this idea on a tiny scale. His idea was taken in the next stage by Church *et al.* and Goldman et al. Up until this point, researchers did not focus on the minimization of error in the sequencing/ storing/ synthesizing steps. So more errors were incorporated in these steps. Also, no efficient error correction algorithms were designed to handle those errors. As a result, they were not able to recover the whole file without manual intervention. Later, Grass *at el.* introduced a wheel mechanism for handling the repetition also they have used a better error correction algorithm, which enabled them to recover the whole file successfully. Erlich *et al.* made the whole system more robust by introducing fountain code, which can handle the dropouts of the sequences. They also maintained equal GC content to reduce the error probability. Errors in the sequence are reduced by avoiding specific sequences as well as improving the sequencing technology. So less redundant sequences are needed to recover the whole file. Which ultimately increases the data density. Rewrite ability and random access are also the fundamental properties of any storage system. Researchers have successfully incorporated these both in larger and smaller scale storage systems. The automation device for synthesis and sequencing by Takahashi at el. opened a whole new door for further research. These continuous improvements over time is a clear indication of the future viability of DNA based storage systems.

References:

[1]J. Bornholt, R. Lopez, D. M. Carmean, L. Ceze, G. Seelig, and K. Strauss, “A DNA-Based Archival Storage System,” in Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems - ASPLOS ’16, 2016, doi: 10.1145/2872362.2872397.

[2] V. Zhirnov, R. M. Zadegan, G. S. Sandhu, G. M. Church, and W. L. Hughes, “Nucleic acid memory,” Nature Materials, vol. 15, no. 4, pp. 366–370, Mar. 2016, doi: 10.1038/nmat4594.

[3] https://www.nrdc.org/sites/default/files/data-center-efficiency-assessment-IB.pdf

[4] N. Goldman et al., “Towards practical, high-capacity, low-maintenance information storage in synthesized DNA,” Nature, vol. 494, no. 7435, pp. 77–80, Jan. 2013, doi: 10.1038/nature11875.

[5] C. T. Clelland, V. Risca, and C. Bancroft, “Hiding messages in DNA microdots,” Nature, vol. 399, no. 6736, pp. 533–534, Jun. 1999, doi: 10.1038/21092.
[6] G. E. Moore, “Cramming More Components Onto Integrated Circuits,” Proceedings of the IEEE, vol. 86, no. 1, pp. 82–85, Jan. 1998.
[7] G. M. Church, Y. Gao, and S. Kosuri, “Next-Generation Digital Information Storage in DNA,” Science, vol. 337, no. 6102, pp. 1628–1628, Aug. 2012, doi: 10.1126/science.1226355.
[8] J. Davis, “Microvenus,” Art Journal, vol. 55, no. 1, p. 70, 1996, doi: 10.2307/777811.
[9] Y. Erlich and D. Zielinski, “DNA Fountain enables a robust and efficient storage architecture,” Science, vol. 355, no. 6328, pp. 950–954, Mar. 2017.
[10] J. J. Schwartz, C. Lee, J. Shendure, Nat. Methods 9, 913–915 (2012).
[11] M. G. Ross et al., Genome Biol. 14, R51 (2013)
[12] G. Ananda, E. Walsh, K. D. Jacob, M. Krasilnikova, K. A. Eckert, F. Chiaromonte, K. D. Makova, Distinct mutational behaviors differentiate short tandem repeats from microsatellites in the human genome. Genome Biol. Evol. 5, 606–620 (2013). doi:10.1093/gbe/evs116 Medline
[13] S. M.Hossein Tabatabaei Yazdi, Yongbo Yuan, Jian Ma, Huimin Zhao, and Olgica Milenkovic. A Rewritable, Random-Access DNA-Based Storage System. Scientific Reports, 5(14138), 2015.
[14] Meinolf Blawat, Klaus Gaedke, Ingo Hu¨tter, Xiao-Ming Chen, Brian Turczyk, Samuel Inverso, Benjamin W Pruitt, and George M Church. Forward Error Correction for DNA Data Storage. Procedia Computer Science, 80:1011–1022, 2016.
[15] Newman, S., Stephenson, A.P., Willsey, M. et al. High density DNA data storage library via dehydration with digital microfluidic retrieval. Nat Commun 10, 1706 (2019). https://doi.org/10.1038/s41467-019-09517-y
[16] Cohen, G. D. & Litsyn, S. Dc-constrained error-correcting codes with small running digital sum. Information Theory, IEEE Transactions on 37, 949–955 (1991).
[17] Morita, H., van Wijngaarden, A. J. & Han Vinck, A. On the construction of maximal prefix-synchronized codes. Information Theory, IEEE Transactions on 42, 2158–2166 (1996).
[18] D. J. C. MacKay, “Fountain codes,” IEE Proceedings - Communications, vol. 152, no. 6, p. 1062, 2005.
[19] R. W. Hamming, “Error Detecting and Error Correcting Codes,” Bell System Technical Journal, vol. 29, no. 2, pp. 147–160, Apr. 1950, doi: 10.1002/j.1538-7305.1950.tb00463.x.
[20] R. C. Bose and D. K. Ray-Chaudhuri, “On a class of error correcting binary group codes,” Information and Control, vol. 3, no. 1, pp. 68–79, Mar. 1960, doi: 10.1016/s0019-9958(60)90287-4.
[21] I. S. Reed and G. Solomon, “Polynomial Codes Over Certain Finite Fields,” Journal of the Society for Industrial and Applied Mathematics, vol. 8, no. 2, pp. 300–304, Jun. 1960, doi: 10.1137/0108018.
[22] S. Ma, I. Saaem, and J. Tian, “Error correction in gene synthesis technology,” Trends in Biotechnology, vol. 30, no. 3, pp. 147–154, Mar. 2012, doi: 10.1016/j.tibtech.2011.10.002.

[23] R. Carlson, “Competition and the Future of Reading and Writing DNA,” in Synthetic Biology, Wiley-VCH Verlag GmbH & Co. KGaA, 2018, pp. 1–13.
[24] Takahashi, C.N., Nguyen, B.H., Strauss, K. et al. Demonstration of End-to-End Automation of DNA Data Storage. Sci Rep 9, 4998 (2019). https://doi.org/10.1038/s41598-019-41228-8
[25] R. N. Grass, R. Heckel, M. Puddu, D. Paunescu, and W. J. Stark, “Robust Chemical Preservation of Digital Information on DNA in Silica with Error-Correcting Codes,” Angewandte Chemie International Edition, vol. 54, no. 8, pp. 2552–2555, Feb. 2015, doi: 10.1002/anie.201411378.
[26] Anavy, L., Vaknin, I., Atar, O. et al. Data storage in DNA with fewer synthesis cycles using composite DNA letters. Nat Biotechnol 37, 1229–1236 (2019). https://doi.org/10.1038/s41587-019-0240-x
[27] Davis, J., Microvenus. Art Journal, 1996. 55(1): p. 70-74.
[28] H. A. Hakami, Z. Chaczko, and A. Kale, “Review of big data storage based on DNA computing,” in Proceedings of the Asia-Pacific Conference on Computer-Aided System Engineering (APCASE ’15), pp. 113–117, Quito, Ecuador, July 2015.
[29] Wetterstrand KA. DNA sequencing costs: data from the NHGRI Genome Sequencing Program (GSP). National Human Genome Research Institute, http://www.genome.gov/sequencingcosts.
[30] Shokralla S, Jennifer L and Spall J et al. Next generation sequencing technologies for environmental DNA research. Mol Ecol 2012; 21: 1794-1805.
[31] De Silva, P. Y., & Ganegoda, G. U. (2016). New Trends of Digital Data Storage in DNA. BioMed Research International, 2016, 1–14. https://doi.org/10.1155/2016/8072463
[32] Behjati, S., & Tarpey, P. S. (2013). What is next generation sequencing? Archives of Disease in Childhood - Education & Practice Edition, 98(6), 236–238. https://doi.org/10.1136/archdischild-2013-304340
[33] Yiming Dong, Fajia Sun, Zhi Ping, Qi Ouyang, Long Qian, DNA storage: research landscape and future prospects, *National Science Review*, , nwaa007, https://doi.org/10.1093/nsr/nwaa007
[34] Kac, E. (1999). Genesis-art of DNA.
[35] Fei, P., & Wang, Z. (2019). LDPC Codes for Portable DNA Storage. 2019 IEEE International Symposium on Information Theory (ISIT). Presented at the 2019 IEEE International Symposium on Information Theory (ISIT). https://doi.org/10.1109/isit.2019.8849814
[36] Ailenberg, M., & Rotstein, O. D. (2009). An improved Huffman coding method for archiving text, images, and music characters in DNA. BioTechniques, 47(3), 747–754. https://doi.org/10.2144/000113218
[37] Organick, L., Ang, S., Chen, Y. et al. Random access in large-scale DNA data storage. Nat Biotechnol 36, 242–248 (2018).
[38] Yazdi, S.M.H.T., Gabrys, R. & Milenkovic, O. Portable and Error-Free DNA-Based Data Storage. Sci Rep 7, 5011 (2017)