A DNA-Based Archival Storage System

James Bornholt† Randolph Lopez† Douglas M. Carmean‡
Luis Ceze† Georg Seelig† Karin Strauss‡

† University of Washington ‡ Microsoft Research

Abstract

Demand for data storage is growing exponentially, but the capacity of existing storage media is not keeping up. Using DNA to archive data is an attractive possibility because it is extremely dense, with a raw limit of 1 exabyte/mm^3 (10^9 GB/mm^3), and long-lasting, with observed half-life of over 500 years.

This paper presents an architecture for a DNA-based archival storage system. It is structured as a key-value store, and leverages common biochemical techniques to provide random access. We also propose a new encoding scheme that offers controllable redundancy, trading off reliability for density. We demonstrate feasibility, random access, and robustness of the proposed encoding with wet lab experiments involving 151 kB of synthesized DNA and a 42 kB random-access subset, and simulation experiments of larger sets calibrated to the wet lab experiments. Finally, we highlight trends in biotechnology that indicate the impending practicality of DNA storage for much larger datasets.

Categories and Subject Descriptors B.3.2 [*Memory Structures*]: Design Styles—Mass storage; J.3 [*Life and Medical Sciences*]: Biology and genetics

Keywords Archival storage; molecular computing; DNA

1. Introduction

The "digital universe" (all digital data worldwide) is forecast to grow to over 16 zettabytes in 2017 [14]. Alarmingly, the exponential growth rate easily exceeds our ability to store it, even when accounting for forecast improvements in storage technologies. A significant fraction of this data is in archival form; for example, Facebook recently built an entire data center dedicated to 1 exabyte of cold storage [18].

ASPLOS '16 April 2–6, 2016, Atlanta, GA, USA

DOI: http://dx.doi.org/10.1145/2872362.2872397

Most of the world's data today is stored on magnetic and optical media [14]. Tape technology has recently seen significant density improvements with tape cartridges as large as 185 TB [25], and is the densest form of storage available commercially today, at about 10 GB/mm^3. Recent research reported feasibility of optical discs capable of storing 1 PB [8], yielding a density of about 100 GB/mm^3. Despite this improvement, storing zettabytes of data would still take millions of units, and use significant physical space. But storage density is only one aspect of archival: durability is also critical. Rotating disks are rated for 3–5 years, and tape is rated for 10–30 years. Current long-term archival storage solutions require refreshes to scrub corrupted data, to replace faulty units, and to refresh technology. If we are to preserve the world's data, we need to seek significant advances in storage density and durability.

Synthetic DNA sequences have long been considered a potential medium for digital data storage [6, 7, 10]. DNA is an attractive possibility because it is extremely dense, with a theoretical limit above 1 EB/mm^3 (eight orders of magnitude denser than tape), and long-lasting, with observed half-life of over 500 years in harsh environments [2]. DNA-based storage also has the benefit of eternal relevance: as long as there is DNA-based life, there will be strong reasons to read and manipulate DNA. The write process for DNA storage maps digital data into DNA nucleotide sequences (a nucleotide is the basic building block of DNA), *synthesizes* (manufactures) the corresponding DNA molecules, and stores them away. Reading the data involves *sequencing* the DNA molecules and decoding the information back to the original digital data. Both synthesis and sequencing are standard practice in biotechnology, from research to diagnostics and therapies.

Progress in DNA storage has been rapid: in 1999, the state-of-the-art in DNA-based storage was encoding and recovering a 23 character message [7]; in 2013, researchers successfully recovered a 739 kB message [6, 10]. This improvement of almost 2×/year has been fueled by exponential reduction in synthesis and sequencing cost and latency; growth in sequencing productivity eclipses even Moore's Law [4]. The volume of data that can be synthesized today is limited mostly by the cost of synthesis and sequencing, but growth in the

biotechnology industry portends orders of magnitude cost reductions and efficiency improvements.

We think the time is ripe to consider DNA-based storage seriously and explore system designs and architectural implications. This paper presents an architecture for a DNA-backed archival storage system, modeled as a key-value store. A DNA storage system must overcome several challenges. First, DNA synthesis and sequencing is far from perfect, with error rates on the order of 1% per nucleotide. Sequences can also degrade while stored, further compromising data integrity. A key aspect of DNA storage is to devise appropriate encoding schemes that can tolerate errors by adding redundancy. Existing approaches have focused on redundancy but have ignored density implications. In this work we propose a new encoding scheme that offers controllable redundancy, enabling different types of data (e.g., text and images) to have different levels of reliability and density. The density of our encoding scheme outperforms existing work while providing similar reliability.

Second, randomly accessing data in DNA-based storage is problematic, resulting in overall read latency that is much longer than write latency. Existing work has provided only large-block access: to read even a single byte from storage, the entire DNA pool must be sequenced and decoded. We propose a method for random access that uses a polymerase chain reaction (PCR) to amplify only the desired data, biasing sequencing towards that data. This design both accelerates reads and ensures that an entire DNA pool need not be sequenced.

We demonstrate the feasibility of our system design with a series of wet lab experiments, in which we successfully stored data in DNA and performed random access to read back only selected values. We further evaluate our design using simulations to understand the error-correction characteristics of different encoding schemes, assess their overheads, and make projections about future feasibility based on technology trends. Our results demonstrate the impending practicality of DNA-based archival storage as a solution to exponential growth in demand for data storage.

2. Background on DNA Manipulation

DNA basics. Naturally occurring DNA consists of four types of nucleotides: adenine (A), cytosine (C), guanine (G), and thymine (T). A DNA strand, or *oligonucleotide*, is a linear sequence of these nucleotides. The two ends of a DNA strand, referred to as the $5'$ and $3'$ ends, are chemically different. DNA sequences are conventionally represented starting with the $5'$ nucleotide end. The interactions between different strands are predictable based on sequence. Two single strands can bind to each other and form a double helix if they are complementary: A in one strand aligns with T in the other, and likewise for C and G. The two strands in a double helix have opposite directionality ($5'$ end binds to the other strand's $3'$ end), and thus the two sequences are the "reverse complement" of each other. Two strands do not need to be fully complementary to bind to one another. Such *partial* complementarity is useful for applications in DNA nanotechnology and other fields, but can also result in undesired "crosstalk" between sequences in complex reaction mixtures containing many sequences.

Selective DNA amplification with polymerase chain reaction (PCR). PCR is a method for exponentially amplifying the concentration of selected sequences of DNA within a pool. A PCR reaction requires four main components: the template, sequencing primers, a thermostable polymerase and individual nucleotides that get incorporated into the DNA strand being amplified. The template is a single- or double-stranded molecule containing the (sub)sequence that will be amplified. The DNA sequencing primers are short synthetic strands that define the beginning and end of the region to be amplified. The polymerase is an enzyme that creates double-stranded DNA from a single-stranded template by "filling in" individual complementary nucleotides one by one, starting from a primer bound to that template. PCR happens in "cycles", each of which doubles the number of templates in a solution. The process can be repeated until the desired number of copies is created.

DNA synthesis. Arbitrary single-strand DNA sequences can be synthesized chemically, nucleotide by nucleotide [15, 17]. The *coupling efficiency* of a synthesis process is the probability that a nucleotide binds to an existing partial strand at each step of the process. Although the coupling efficiency for each step can be higher than 99%, this small error still results in an exponential decrease of product yield with increasing length and limits the size of oligonucleotides that can be efficiently synthesized to about 200 nucleotides. In practice, synthesis of a given sequence uses a large number of parallel start sites and results in many truncated byproducts (the dominant error in DNA synthesis), in addition to many copies of the full length target sequence. Thus, despite errors in synthesizing any specific strand, a given synthesis batch will usually produce many perfect strands. Moreover, modern array synthesis techniques [15] can synthesize complex pools of nearly 10^5 different oligonucleotides in parallel.

DNA sequencing. There are several high-throughput sequencing techniques, but the most popular methods (such as that used by Illumina) use DNA polymerase enzymes and are commonly referred to as "sequencing by synthesis". The strand of interest serves as a template for the polymerase, which creates a complement of the strand. Importantly, *fluorescent* nucleotides are used during this synthesis process. Since each type of fluorescent nucleotide emits a different color, it is possible to read out the complement sequence optically. Sequencing is error-prone, but as with synthesis, in aggregate, sequencing typically produces enough precise reads of each strand.

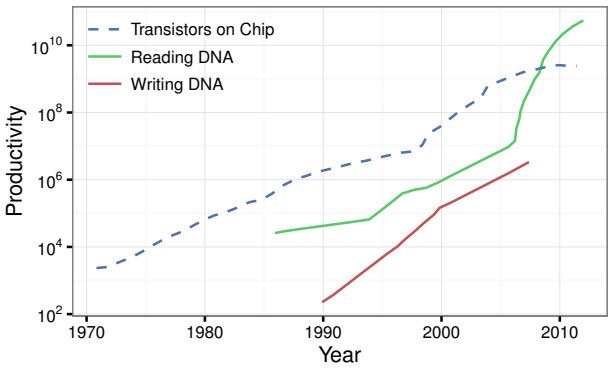


Figure 1. Carlson curves [4]: trends in DNA synthesis and sequencing technology compared to Moore's Law. DNA productivity is measured in nucleotides per person per day. Recent growth in sequencing technology eclipses Moore.

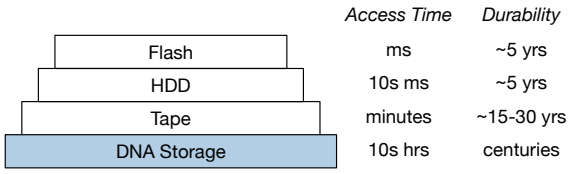


Figure 2. DNA storage as the bottom level of the storage hierarchy

Sequencing and synthesis improvement projections. Today, neither the performance nor the cost of DNA synthesis and sequencing is viable for data storage purposes. However, they have historically seen exponential improvements. Their cost reductions and throughput improvements have been compared to Moore's Law in Carlson's Curves [4], as shown in Figure 1. It shows that sequencing productivity has been growing faster than Moore's Law. Important biotechnology applications such as genomics and the development of smart drugs are expected to continue driving these improvements, eventually making data storage a viable application.

3. A DNA Storage System

We envision DNA storage as the very last level of a deep storage hierarchy, providing very dense and durable archival storage with access times of many hours to days (Figure 2). DNA synthesis and sequencing can be made arbitrarily parallel, making the necessary read and write bandwidths attainable. We now describe our proposal of a system for DNA-based storage with random access support.

3.1 Overview

A DNA storage system consists of a DNA synthesizer that encodes the data to be stored in DNA, a storage container with compartments that store pools of DNA that map to a volume, and a DNA sequencer that reads DNA sequences and converts them back into digital data. Figure 3 shows an overview of the integrated system.

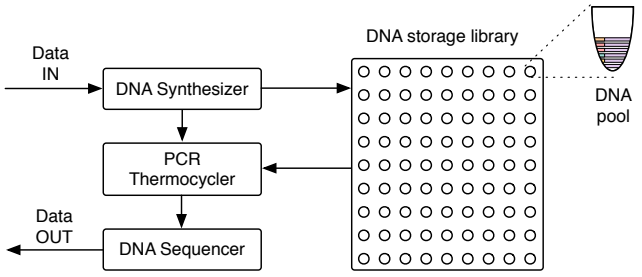


Figure 3. Overview of a DNA storage system.

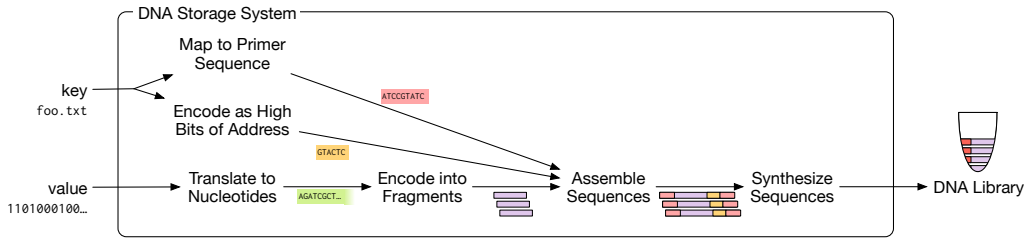
The basic unit of DNA storage is a DNA strand that is roughly 100-200 nucleotides long, capable of storing 50-100 bits total. Therefore, a typical data object maps to a very large number of DNA strands. The DNA strands will be stored in "pools" that have stochastic spatial organization and do not permit structured addressing, unlike electronic storage media. Therefore, it is necessary to embed the address itself into the data stored in a strand. This way, after sequencing, one can reassemble the original data value. We discuss digital data representation in DNA in Section 4.

3.2 Interface and Addressing

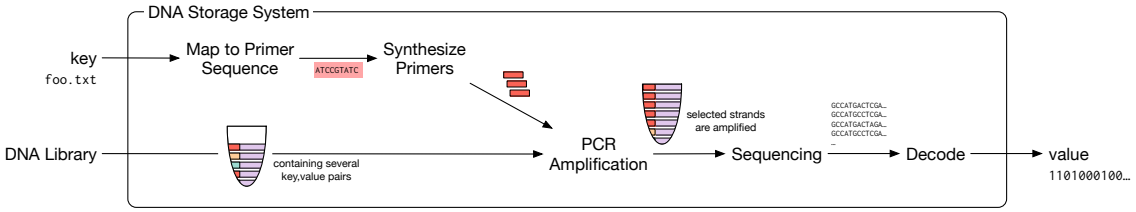
A storage system needs a way to assign identification tags to data objects so they can be retrieved later. We choose a simple key-value architecture, where a `put(key, value)` operation associates `value` with `key`, and a `get(key)` operation retrieves the `value` assigned to `key`. To implement a key-value interface in a DNA storage system, we need: (1) a function that maps a key to the DNA pool (in the library) where the strands that contain data reside; and (2) a mechanism to selectively retrieve only desired portions of a pool (i.e, random access), since the DNA container will likely store significantly more data than the desired object.

We implement random access by mapping a key to a pair of PCR primers. At write time, those primers are added to the strands. At read time, those same primers are used in PCR to amplify only the strands with the desired keys. Because the resulting pool will have a much higher concentration of the desired strands, a sample from that pool is very likely to contain all of those strands.

Separating the DNA strands into a collection of pools (Figure 3) balances a trade-off between storage density, reliability, and performance. The most dense way to store data would be to have all strands in a single pool, but this arrangement sacrifices reliability for two reasons. First, a single pool requires many different primers to distinguish all keys, which increases the chance that two primers react poorly to each other. Second, a single pool reduces the likelihood that a random sample drawn during the read process will contain all the desired data. On the other hand, using a separate pool per key sacrifices density excessively. We therefore use a library of reasonably-sized pools, and use random access within each pool.



(a) The write process performs `put(key, value)`, generating a DNA library.



(b) The read process performs `get(key)` on a DNA library, returning the value.

Figure 4. Overview of a DNA storage system operation as a key-value store.

3.3 System Operation

Figure 4 shows flowcharts for the write (`put`) and read (`get`) processes in more detail. The write process (Fig. 4(a)) takes as input the key and value to store. It uses the key to obtain the PCR primer sequences, compute the high part of the address, and to determine the pool in the DNA library where the resulting strands will be stored. The low part of the address indexes the multiple strands generated by chunking the value (see Sec. 4.2). Next, it encodes the data addresses, payloads, and error detection codes, and attaches the primer target sequences, to produce final DNA sequences for the synthesizer to manufacture. The resulting DNA molecules are stored in the storage library for archival.

The read process (Fig. 4(b)) takes as input a key. It uses the key to obtain the PCR primer sequences that identify molecules in the pool associated with that key. Next, the storage system physically extracts a sample from the DNA pool that contains the stored data, but likely also includes large amounts of unrelated data. The sample and the PCR primers are sent to the PCR thermocycler, which amplifies only the desired strands. The resulting pool goes to the DNA sequencer, which ultimately produces the digital data readout. Note that this process might be iterative since it may require multiple samples and sequencing steps to extract all the data associated with the desired keys. The DNA synthesizer is used for both producing the DNA strands that hold data payload as well as synthesizing the PCR primers used to amplify data during the random access read process.

The read process removes a sample of DNA from the pool, and so cumulative reads reduce the quantity of DNA available for future operations. But DNA is easy to replicate, and so the pools can easily be replenished after read operations if necessary. If successive amplification is problematic, a pool can also be completely resynthesized after a read operation.

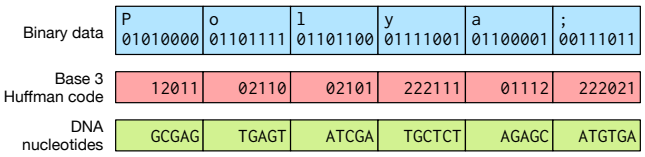
4. Representing Data in DNA

While DNA has many properties that make it different from existing storage media, there are parallels between traditional storage and DNA storage. At the lowest levels, traditional storage media store raw bits. The storage device abstracts the physical media, which could be magnetic state, or the charge in a capacitor, or the stable state of a flip-flop, and presents to the storage hierarchy raw digital data. In a similar way, the abstraction of DNA storage is the nucleotide: though a nucleotide is an organic molecule consisting of one base (A, C, G, or T) and a sugar-phosphate backbone, the abstraction of DNA storage is as a contiguous string of quaternary (base-4) numerals. This section describes the challenges of representing data in DNA, and presents several encodings that overcome these challenges.

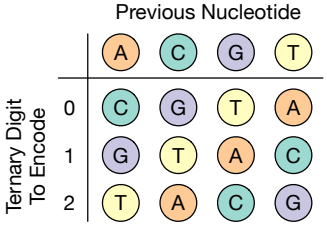
4.1 Representation

The obvious approach to store binary data in DNA is to encode the binary data in base 4, producing a string of $n/2$ quaternary digits from a string of n binary bits. The quaternary digits can then be mapped to DNA nucleotides (e.g., mapping 0, 1, 2, 3 to A, C, G, T, respectively). For example, the binary string 01110001 maps to the base-4 string 1301, and then to the DNA sequence CTAC. However, DNA synthesis and sequencing processes are prone to a wide variety of errors (substitutions, insertions, and deletions of nucleotides), requiring a more careful encoding.

The likelihood of some forms of error can be reduced by encoding binary data in base 3 instead of base 4 [10], as Figure 5(a) illustrates. Each ternary digit maps to a DNA nucleotide based on a rotating code (Figure 5(b)) that avoids repeating the same nucleotide twice. This encoding avoids *homopolymers*—repetitions of the same nucleotide that significantly increase the chance of sequencing errors [20].



(a) Translating binary data to DNA nucleotides via a Huffman code.



(b) A rotating encoding to nucleotides avoids homopolymers (repetitions of the same nucleotide), which are error-prone.

Figure 5. Encoding a stream of binary data as a stream of nucleotides. A Huffman code translates binary to ternary digits, and a rotating encoding translates ternary digits to nucleotides.

Because base 3 is not a multiple of base 2, mapping directly between the bases would be inefficient: 6 ternary digits ($3^6 = 729$) can store 9 bits of data ($2^9 = 512$), but waste 217 possible states. Instead, we use a Huffman code [13] that maps each binary byte to either 5 or 6 ternary digits. For example, the Huffman code maps the binary string 01100001 to the base-3 string 01112. The rotating nucleotide encoding maps this string to the DNA sequence CTCTG. The code maps more common ASCII characters to 5 digit strings, offering minor compression benefits for textual data, though the effect on overall storage density is insignificant.

4.2 Data Format

Another practical issue with representing data in DNA is that current synthesis technology does not scale beyond sequences of low hundreds of nucleotides. Data beyond the hundreds of bits therefore cannot be synthesized as a single strand of DNA. In addition, DNA pools do not offer spatial isolation, and so a pool contains data for many different keys which are irrelevant to a single read operation. Isolating only the molecules of interest is non-trivial, and so existing DNA storage techniques generally sequence the entire solution, which incurs significant cost and time overheads.

To overcome these two challenges, we organize data in DNA in a similar fashion to Goldman et al. [10], as shown in Figure 6. Segmenting the nucleotide representation into blocks, which we synthesize as separate strands, allows storage of large values. Tagging those strands with identifying primers allows the read process to isolate molecules of interest and so perform random access. Below we describe these designs in detail.

Payload. The string of nucleotides representing the data to be stored is broken into data blocks, whose length depends on the desired strand length and the additional overheads

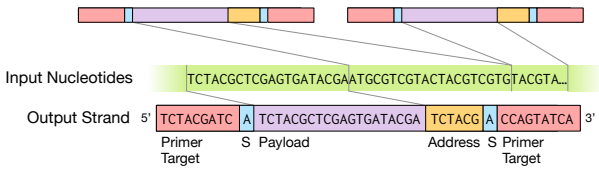


Figure 6. An overview of the DNA data encoding format. After translating to nucleotides, the stream is divided into strands. Each strand contains a payload from the stream, together with addressing information to identify the strand and primer targets necessary for PCR and sequencing.

of the format. To aid decoding, two sense nucleotides ("S" in Figure 6) indicate whether the strand has been reverse complemented (this is done to avoid certain pathological cases).

Address. Each data block is augmented with addressing information to identify its location in the input data string. The address space is in two parts. The high part of the address identifies the key a block is associated with. The low part of the address indexes the block within the value associated with that key. The combined address is padded to a fixed length and converted to nucleotides as described above. A parity nucleotide is added for basic error detection.

Primers. To each end of the strand, we attach primer sequences. These sequences serve as a "foothold" for the PCR process, and allow the PCR to selectively amplify only those strands with a chosen primer sequence.

Random Access. We exploit primer sequences to provide random access: by assigning different primers to different strands, we can perform sequencing on only a selected group of strands. Existing work on DNA storage uses a single primer sequence for all strands. While this design suffices for data recovery, it is inefficient: the entire pool (i.e., the strands for every key) must be sequenced to recover one value.

To provide random access, we instead design a mapping from keys to unique primer sequences. All strands for a particular object share a common primer, and different strands with the same primer are distinguished by their different addresses.

Primers allow random access via a polymerase chain reaction (PCR), which produces many copies of a piece of DNA in a solution. By controlling the sequences used as primers for PCR, we can dictate which strands in the solution are amplified. To read a particular key's value from the solution, we simply perform a PCR process using that key's primer, which amplifies the selected strands. The sequencing process then reads only those strands, rather than the entire pool. The amplification means sequencing can be faster and cheaper, because the probability of recovering the desired object is higher.

Note that not all adapters and primers have the same behavior or effectiveness during PCR. Also, the actual sequences affect the PCR cycle temperatures. Discussing adapter and

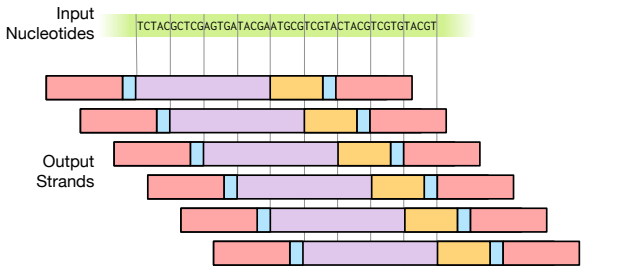


Figure 7. An encoding proposed by Goldman et al. [10]. The payloads of each strand are overlapping segments of the input stream, such that each block in the stream appears in four distinct strands.

primer design is outside the scope of this paper. The hash function that maps addresses to primers can be implemented as a table lookup of primers that are known to work well and have known thermocycling temperatures. For this paper, we used primer designs from prior work [24].

5. Encodings for Reliable Storage

The previous sections described the organization of a DNA-based storage system, and in particular, how data can be broken into strands of DNA. However, the encoding implied by the previous section is naive: each bit of binary data is encoded in exactly one location in the output DNA strands, and so relies on the robustness of DNA for durability. A more robust design would provide redundancy at the data encoding level. This section discusses existing work on encodings for redundancy, highlights the reliability–density trade-off implicit in encoding design, and presents a new encoding with similar reliability to and yet higher density than existing work.

5.1 Existing Encodings

Early work in DNA storage used encodings simpler than the one we describe above. For example, Bancroft et al. [3] translate text to DNA by means of a simple ternary encoding: each of the 26 English characters and a space character maps to a sequence of three nucleotides drawn from A, C, and T (so exactly $3^3 = 27$ characters can be represented). The authors successfully recovered a message of 106 characters, but this encoding suffers substantial overheads and poor reliability for longer messages.

Goldman encoding. We focus on an existing encoding proposed by Goldman et al. [10], shown in Figure 7. This encoding splits the input DNA nucleotides into overlapping segments to provide fourfold redundancy for each segment. Each window of four segments corresponds to a strand in the output encoding. The authors used this encoding to successfully recover a 739 kB message. We use this encoding as a baseline because it is, to our knowledge, the most successful published DNA storage technique. In addition, it offers a tunable level of redundancy, by reducing the width of the segments and therefore repeating them more often in strands of the same length (for example, if the overlapping segments were half as long as in Figure 7, they would be repeated in eight strands instead of four).

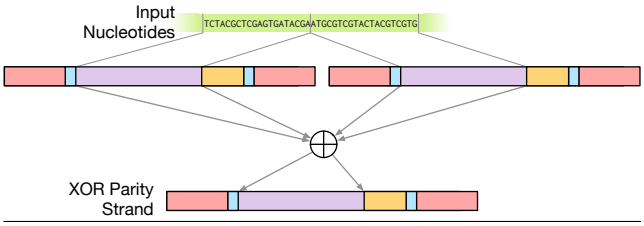


Figure 8. Our proposed encoding incorporates redundancy by taking the exclusive-or of two payloads to form a third. Recovering any two of the three strands is sufficient to recover the third.

5.2 XOR Encoding

While the Goldman encoding provides high reliability, it also incurs significant overhead: each block in the input string is repeated four times. We propose a simple new encoding that provides similar levels of redundancy to prior work, but with reduced overhead.

Our encoding, shown in Figure 8, provides redundancy by a simple exclusive-or operation at the strand level. We take the exclusive-or $A \oplus B$ of the payloads A and B of two strands, which produces a new payload and so a new DNA strand. The address block of the new strand encodes the addresses of the input strands that were the inputs to the exclusive-or; the high bit of the address is used to indicate whether a strand is an original payload or an exclusive-or strand. This encoding provides its redundancy in a similar fashion to RAID 5: any two of the three strands A, B, and $A \oplus B$ are sufficient to recover the third.

The reliability of this encoding is similar to that of Goldman. In Section 6, we show that we successfully recovered objects from both encodings in a wet lab experiment. However, the theoretical density of this encoding is much higher than Goldman—where in their encoding each nucleotide repeats (up to) four times, in ours each nucleotide repeats an average of 1.5 times. In practice, the density difference is lower, due to the overheads of addressing and primers that are constant between the two encodings. Simulation results in Section 7 show our encoding to be twice as dense as that of Goldman, and for all practical DNA synthesis and sequencing technologies, it provides equivalent levels of reliability.

5.3 Tunable Redundancy

Recent work in approximate storage [12, 23] shows that many applications do not need high-precision storage for every data structure. For example, while the header data of a JPEG file is critical to successful decoding, small errors in the payload are tolerable at the cost of some decoding imprecision.

One key advantage of our encoding is that the level of redundancy is tunable at a per-block granularity. For critical data, we can provide high redundancy by pairing critical

blocks with many other blocks: if A is critical, produce $A \oplus B$, $A \oplus C$, etc. On the other hand, for blocks that are less critical, we can further reduce their redundancy: instead of including only two blocks in an exclusive-or, we can include n, such that any $n-1$ of the n blocks is sufficient to recover the last, at an average density overhead of $1/n$.

In addition to improving density, tunable redundancy has a significant effect on performance. Both DNA synthesis and sequencing are slower and more error-prone with larger datasets, and this error does not always grow linearly in size. It is often economically viable to synthesize smaller DNA pools with more accurate technology, while larger pools are out of reach. Tunable redundancy allows the storage system to optimize the balance between reliability and efficiency.

5.4 Factors in Encoding Design

As with many encodings in the literature, ours ignores special properties of DNA synthesis and sequencing that make choosing an optimal encoding more complex. In particular, errors in synthesis and sequencing are not uniform: they vary according to location within a strand, and different sequences of nucleotides are more susceptible to undesired reactions.

Synthesis error grows with the length of the strand being synthesized; nucleotides towards the end of the strand are more likely to be incorrect. Some of these errors are easily ignored, because they result in truncations, where the product strand is not of the correct length. Other errors are more insidious, with substitution errors being particularly common.

An improved version of our encoding would tolerate variable substitution errors by not aligning the strands directly. For example, rather than computing $A \oplus B$, we might instead compute $A \oplus B'$, where B' is B in reverse. Since nucleotides at the end of a strand are more error-prone, reversing one strand ensures that the average quality is (roughly) constant along the strand, so that each nucleotide has at least some redundancy information stored in a high-reliability position.

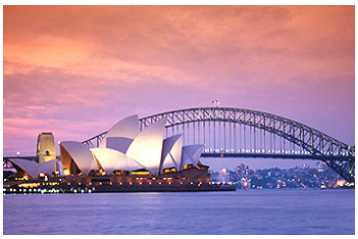
6. Experiments

To demonstrate the feasibility of DNA storage with random access capabilities, we encoded four image files using the two encodings described in Section 5. The files varied in size from 5 kB to 84 kB. We synthesized these files and sequenced the resulting DNA to recover the files. This section describes our experience with the synthesis and sequencing process, and presents results demonstrating that DNA storage is practical and that random access works. We used the results of our experiments to inform the design of a simulator, which we use in Section 7 to perform more experiments exploring the design space of data encoding and durability.

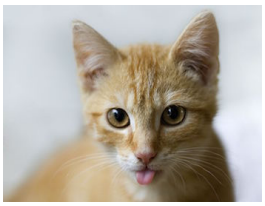
6.1 Materials and Method

This section briefly describes the experimental protocol; see Appendix A for details.

We used as input to the storage system four image files. For each image file `x.jpg`, we generated DNA sequences



(a) `sydney.jpg`, 24301 bytes



(b) `cat.jpg`, 11901 bytes

(c) `smiley.jpg`, 5665 bytes

Figure 9. Three image files we synthesized and sequenced for our experiments.

corresponding to the output of `put(x.jpg, ...)`. We performed these operations using two different encodings – the Goldman encoding and our proposed XOR encoding – described in Section 5. Combined, the eight operations produced 45,652 sequences of length 120 nucleotides, representing 151 kB of data.

To demonstrate that DNA storage allows effective random access, we performed four `get` operations: selecting three of the four files encoded with the Goldman encoding, and one of the four encoded with the XOR encoding. The three files retrieved from the Goldman encoding are in Figure 9; we also performed `get(sydney.jpg)` on the XOR-encoded data.

The synthesized sequences were prepared for sequencing by amplification via the polymerase chain reaction (PCR) method. The product was sequenced using an Illumina MiSeq platform. The selected `get` operations total 16,994 sequences and 42 kB. Sequencing produced 20.8 M reads of sequences in the pool. We inspected the results and observed no reads of sequences that were not selected – so random access was effective in amplifying only the target files.

6.2 Results

File Recovery. We successfully recovered all four files from the sequenced DNA. Three of the files were recovered without manual intervention. One file – `cat.jpg` encoded with the Goldman encoder – incurred a one-byte error in the JPEG header, which we fixed by hand. As described, the design of the Goldman encoder provides no redundancy for the first and last bytes of a file, and so this error was due to random substitution in either sequencing or synthesis. We could mitigate this error scenario by trivially extending that algorithm to wrap the redundant strands past the end of the file and back to the beginning.

Sequencing Depth. Figure 10 shows the distribution of sequencing depths over the 16,994 selected sequences. The sequencing depth of a strand is the number of times it was perfectly sequenced. Of the 20.8 M reads from the sequencing

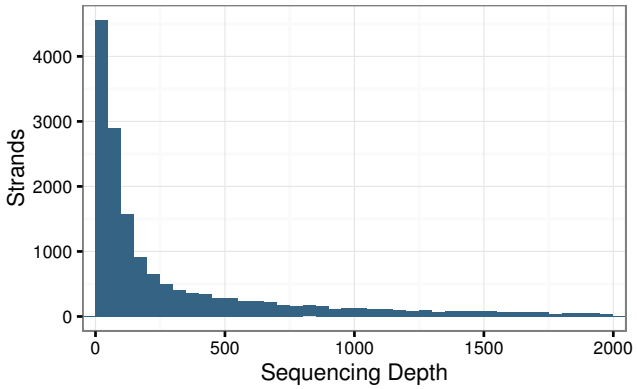


Figure 10. Distribution of sequencing depths over all encoded strands.

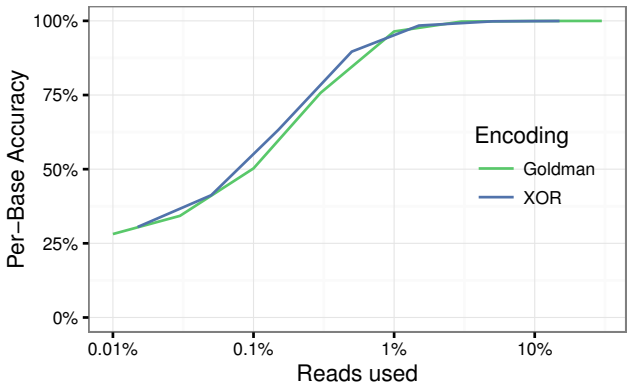


Figure 11. Decoding accuracy as a function of sequencing depth.

run, 8.6 M were error-free reads of a strand in the desired pool. The distribution of reads by sequence is heavily skewed with a mean depth of 506 and median depth of 128. These results suggest that encodings need to be robust not only to missing sequences (which get very few reads), but heavily amplified incorrect sequences.

Reduced Sequencing Depth. The sequencing depth achieved in our experiment is more than sufficient to recover the encoded data. Sequencing technology can reduce sequencing depth in exchange for faster, higher-throughput results. To determine whether our encodings are still effective as sequencing depth reduces, we randomly subsampled the 20.8 M reads we achieved and tried to decode `sydney.jpg` again, using both the Goldman and XOR encodings.

Figure 11 shows that both encodings respond similarly to reduced sequencing depth. The x-axis plots the fraction of the 20.8 M reads used, and the y-axis the accuracy of the decoded file. Both encodings tend to 25% accuracy as the depth reduces, as both decoders randomly guess one of the four nucleotides if no data is available. The accuracy of the two encodings is similar; however, the XOR encoding is higher density than the Goldman encoding.

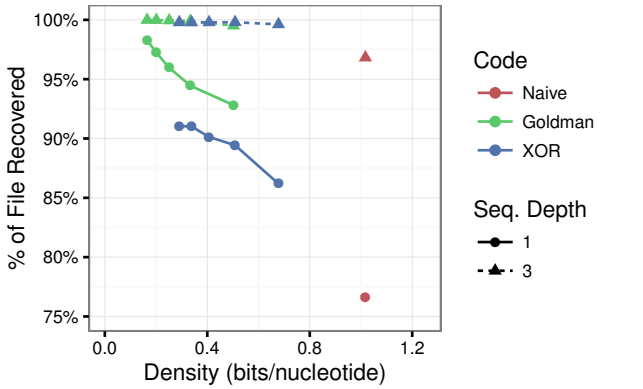


Figure 12. Reliability of encoded data as a function of storage density at different sequencing depths.

Naive Encoding. Figure 8 shows that the XOR encoding is actually a superset of a naive encoding: if we ignore strands which are products $A \oplus B$, we are left with only the naively encoded strands A and B. We attempted to decode `sydney.jpg` while ignoring the XOR products. We found that 11 strands were missing entirely, and even after improving the decoder to arbitrarily guess the values of missing strands, were not able to recover a valid JPEG file. The XOR encoding corrected all these errors at a lower density overhead than the Goldman encoding. These results suggest that even at very high sequencing depths, a naive encoding is not sufficient for DNA storage: encodings must provide their own robustness to errors.

7. Simulation

We used the results of the experiments in Section 6 to inform the design of a simulator for DNA synthesis and sequencing. The simulator allows experimenting with new encodings and new configurations for existing encodings. This section uses the simulator to answer two questions about DNA storage: first, how do different encodings trade storage density for reliability, and second, what is the effect of decay on the reliability of stored data?

Reliability and Density. The encodings we described in Section 5 can be reconfigured to provide either higher density or higher reliability. To examine this trade-off between different encodings, we encoded the `sydney.jpg` file (Fig. 9) with a variety of configurations. These configurations vary the number of strands where a piece of data is included, by changing the overlap between strands for Goldman and increasing the number of XOR copies for XOR.

Figure 12 plots the density achieved by an encoding (x-axis) against decoding reliability (y-axis). The density is calculated as the file size divided by the total number of bases used to encode the file. Figure 12 includes three different encoding mechanisms: a naive encoding with no redundancy, the encoding proposed by Goldman, and our proposed XOR encoding. It presents the results for two

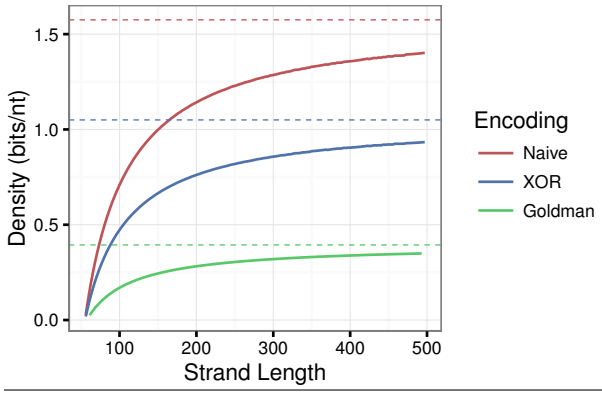


Figure 13. Density of different encodings as a function of the length of each synthesized DNA strand. Dashed lines are asymptotes for infinite-length strands.

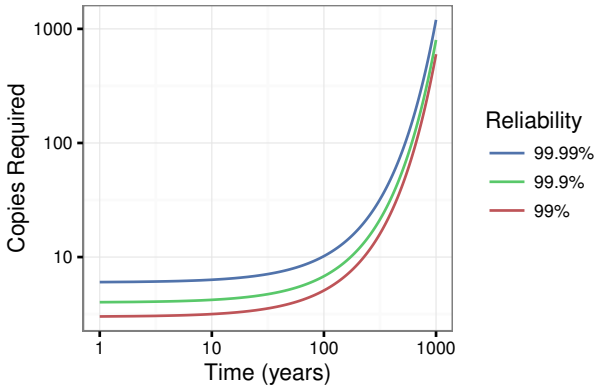


Figure 14. Average number of copies of sequences required to ensure a desired reliability over time.

sequencing depths, 1 and 3. Naive encoding has the lowest reliability because there is no redundancy. As the sequencing depth increases, the reliability improves, but as observed in the wet lab experiments, even at higher sequencing depths, the Naive encoding is not sufficient to provide full data recovery. For both tunable encodings, additional redundancy increases robustness, but affects density negatively. For sequencing depth of 1, where only a single copy of each strand is available, any error causes information loss if no redundancy is available (red dot). As more redundancy is added (blue and green curves), the encoding becomes more resilient to errors. At sequencing depth 1 and same density, Goldman is more resilient than XOR because it does not combine then replicate bits, it simply replicates them. As sequencing depths increase, XOR becomes as reliable as Goldman because the probability of having no copies at all of the original data lowers significantly.

Density and Strand Length. One limiting factor for DNA storage is strand length: current DNA synthesis technology can only viably produce strands of lengths less than 200, and our wet lab experiments (Sec. 6) used strands of length 120. But future synthesis technology promises to increase this limit, as many fields of biology require longer artificial DNA sequences (for example, the average human gene has on the order of 10^4 nucleotides).

Figure 13 plots the density of different encodings on the y-axis as a function of the strand length on the x-axis. As strand length grows, addressing and other overheads become less significant, and density becomes a function only of the encoding. In the limit, the density of our XOR encoding is 2.6× that of Goldman, and yet our results show similar reliability. The XOR encoding is also two-thirds the density of a naive encoding, which suffers much worse error rates.

Decay. Finally, we used the simulator to evaluate the durability of DNA storage over time. The durability of electronic storage is, in the best case, in the tens of years. In contrast, the half life of single stranded DNA is much longer. To demonstrate the durability of DNA storage, we simulated decay at room temperature, according to rates observed in recent work [11]. Storage at lower temperatures significantly increases the half life of the stored DNA.

Figure 14 shows desired timespan on the x-axis and the number of copies of each strand required to achieve a desired reliability after the timespan on the y-axis. Different curves correspond to different desired reliabilities. For example, the 99.99% reliability line says that to have a 99.99% chance of recovering an encoded value after 100 years, we need to store only 10 copies of the strands for that value. The results show that even very few copies are sufficient to provide high reliability long beyond the half lives of existing electronic media. In summary, high reliability for long time intervals does not have much impact on DNA storage density.

8. Discussion and Future Work

The results in Sections 6 and 7 demonstrate a full DNA storage system and its advantages. This section provides a more in-depth examination of experimental data and discusses potential improvements.

The Real Source of Errors. The results in Section 6 showed that error rates in a complete storage pipeline are high enough to require redundant encoding. This section shows where these errors come from, so that we can decide where to focus our attention when attempting to reduce errors.

To answer this question, we synthesized a small subset of strands using a high fidelity on-column synthesis process (Single). In this process, each sequence is synthesized individually resulting in low error rate, but high cost and low throughput incompatible with information storage applications. In contrast, the results in Section 6 were generated using a high-throughput microarray-based synthesis process (Array) that is not high fidelity, but is currently among the lowest cost DNA synthesis alternatives.

Figure 15 compares the overall error distribution for identical strands obtained through Single and Array synthesis as a function of the relative position within the strand. The

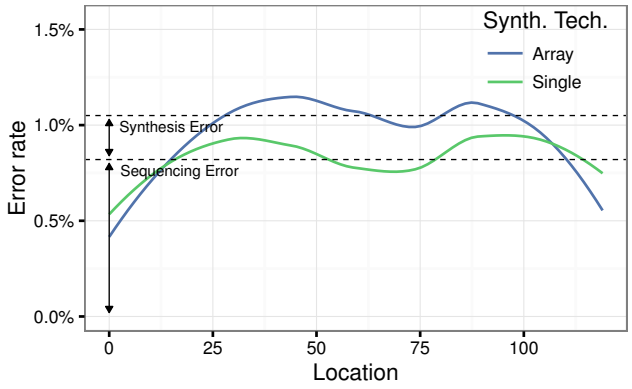


Figure 15. Distribution of DNA errors from two synthesis technologies. Because Single synthesis is effectively error-free, its results reflect sequencing error alone.

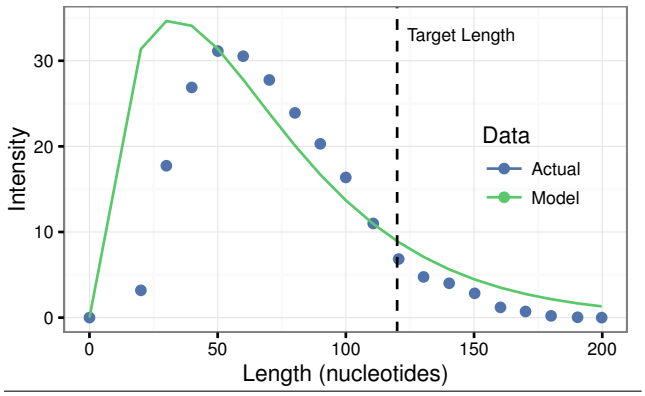


Figure 16. Expected and observed distributions of strand length from DNA synthesis. Sequencing recovers only strands of the target length, but most of the synthesis product is not full length.

overall error rate is due to both sequencing and synthesis errors. However, if we assume that the error rate for Single synthesis is near zero, we can interpret errors measured for those strands as solely due to sequencing. The difference between the error rates for the stands derived from Single and Array can then be interpreted as synthesis errors. We clearly see that sequencing error dominates for all locations: the sequencing error rate is on average an order of magnitude higher. These results show that future technology should focus on improving the accuracy of sequencing.

Synthesis Efficiency. In addition to incorrect strands, micro-array synthesis can produce *truncated* strands. Standard sequencing processes exclude these truncated strands, so they are unused for data recovery and represent waste.

To illustrate the potential losses due to truncation, Figure 16 shows the length distribution of the synthesized strands, as determined by gel electrophoresis (see Appendix A for details). Less than 5% of the pool is of the target length of 120 nucleotides – other strands are either truncated early, or (much more rarely) made longer. This indicates that work in reducing number of fragments could improve synthesis costs by up to about one order of magnitude.

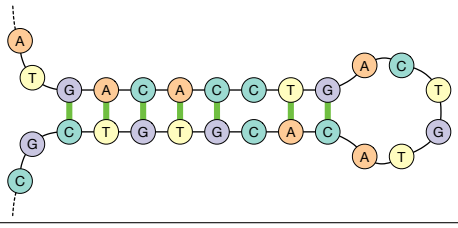


Figure 17. A hairpin, in which a single sequence of DNA binds to itself because nearby regions are self-complementary. Hairpins make DNA amplification and sequencing more error prone.

Avoiding Bad Sequences. The representation we (and others) have used does not avoid more complex sources of error in DNA data representation. For example, Figure 17 shows a *hairpin*, in which a single sequence of DNA binds to itself, folding its two ends together because they are (partially) complementary. The binding of the two ends to each other prevents this strand from being easily amplified and sequenced. A more robust representation would avoid creating sequences that are self-complementary to reduce the chance of this self-hybridization. Of course, restricting self-complementarity also reduces the potential density of the representation, so it presents a trade-off between density and reliability.

Similarly, if different strands are partially complementary, there is a chance they will bind to each other. A more robust encoding would try to mitigate this chance. For example, the mapping in Figure 5(b) from ternary digits to nucleotides need not be static. Instead, it could be selected on a per-strand basis by selecting the encoding that produces the least self-complementary and partially complementary strands.

We intend to explore these directions as future work, but thus far these issues have had little effect on our experiments.

9. Related Work

Encoding data in DNA has a long line of research in the biology community. Early examples encoded and recovered very short messages: Clelland et al. recovered a 23 character message in 1999 [7], and Leier et al. recover three 9-bit numbers in 2000 [16]. The first significant scaling improvements were made by Gibson et al. in 2010, successfully recovering 1280 characters encoded in a bacterial genome as "watermarks" [9] – but note this approach is *in vivo* (inside an organism), whereas ours is *in vitro* (outside), so the technology is very different and inapplicable to large-scale storage. More scaling improvements were made by Church et al. in 2012, who recovered a 643 kB message [6], and Goldman et al. recovered a 739 kB message also in 2012 [10]. However, both these results required manual intervention: Church et al. had to manually correct ten bits of error, and Goldman et al. lost two sequences of 25 nucleotides.

Most recently, Grass et al. recovered an 83 kB message without error [11]. Their design uses a Reed-Solomon code [22], striping the entire dataset across 5000 DNA strands. While this design leads to excellent redundancy, it defeats the desire for random access.

Concurrent with our work, Yazdi et al. [28] developed a method for rewritable random-access DNA-based storage. Its encoding is dictionary-based and focuses on storage of text, while our approach accommodates arbitrary binary data. We do not support rewritability, which adds substantial complexity, because write-once is appropriate for archival items (e.g., photos) that are unlikely to change. If necessary, one can use a log-based approach for rewriting objects, since the density of DNA exceeds the overhead of logging.

The use of DNA as a computing substrate also has a long history: in 1994, Adleman proposed composing Hamiltonian paths with DNA [1]. Researchers have proposed the use of DNA for Boolean circuits [26], neural networks [21], and chemical reaction networks [5]. DNA computing has also begun making inroads in the architecture community: Muscat et al. explore the architectural challenges of DNA strand-displacement circuits [19], and Talawar examines the design of a DNA-based crossbar interconnection network [27].

10. Conclusion

DNA-based storage has the potential to be the ultimate archival storage solution: it is extremely dense and durable. While this is not practical yet due to the current state of DNA synthesis and sequencing, both technologies are improving at an exponential rate with advances in the biotechnology industry. Given the impending limits of silicon technology, we believe that hybrid silicon and biochemical systems are worth serious consideration: time is ripe for computer architects to consider incorporating biomolecules as an integral part of computer design. DNA-based storage is one clear example of this direction. Biotechnology has benefited tremendously from progress in silicon technology developed by the computer industry; perhaps now is the time for the computer industry to borrow back from the biotechnology industry to advance the state of the art in computer systems.

Acknowledgements

We thank Brandon Holt, Emina Torlak, Xi Wang, and the anonymous reviewers for their feedback on earlier versions of this work; Robert Carlson for providing historic information on DNA synthesis and sequencing; and the members of the Sampa, Molecular Information Systems, and Seelig labs at UW, and the Emerging Storage and Computation team at Microsoft Research, for their support of this project. This work was supported in part by NSF grants #1064497 and #1409831, gifts by Microsoft Research, and by the David Notkin Endowed Graduate Fellowship.

A. Appendix: Materials and Method

Primers for the PCR reaction (Fig. 18) were designed to amplify specific files and also to incorporate sequence domains that are necessary for sequencing. Each primer incorporated overhangs that included three sequence domains in addition to the amplification domain necessary for PCR amplification. The first domain included the sequences necessary for binding to the Illumina flow cell during next generation sequencing. The second domain included a custom sequencing-priming region designed for the sequencing primer to bind. This region allows for sequencing of multiple files in the same sequencing run since the sequencing primer region becomes independent of the oligonucleotide pool. These sequences were generated using Nupack [29], software for thermodynamic analysis of interacting nucleic acid strands, in order to avoid the formation of secondary structure that could interfere with the PCR reaction. The third domain consisted of a 12-nucleotide long degenerate region intended to optimize cluster detection in the Illumina sequencing platform.

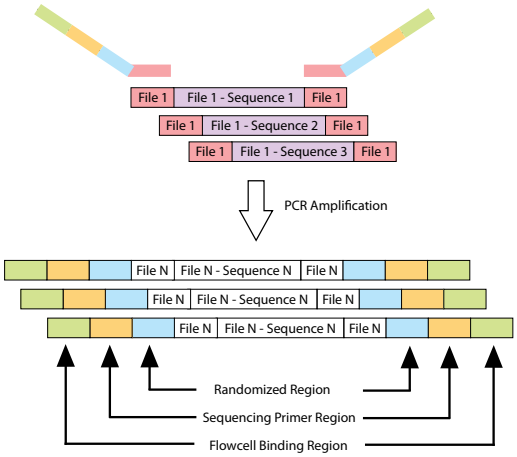


Figure 18. PCR amplifies the strands in the file, and attaches new regions to each end that allow for sequencing.

PCR amplification was performed using Platinum® PCR SuperMix High Fidelity MasterMix from Life Technologies. The cycling conditions were (i) 95°C for 3 min, (ii) 95°C for 20 s, (iii) 55°C for 20 s, (iv) 72°C for 160 s, and (v) looping through (ii)–(iv) 30 times. The PCR amplification output was purified via gel extraction and quantified before next generation sequencing. Finally, the product was sequenced using an Illumina MiSeq sequencing platform.

Synthesis Efficiency. To determine the results in Figure 16, which presents a distribution of strand lengths produced by synthesis, we designed a mathematical model to estimate the nucleotide *coupling efficiency*, which is the probability that a nucleotide will be added to the strand during each of the 120 coupling cycles in the synthesis process. The model says that the likelihood of observing a strand of length n is proportional to

$$Intensity = nN_t p^n (1-p)$$

where N_t is the total number of DNA molecules being synthesized in the array, p is the coupling efficiency, and *Intensity* is the observed flouresence measured from the gel electrophoresis shown in Figure 19. By fitting this model, we estimated the nucleotide coupling efficiency in the synthesis process to be approximately 0.975.

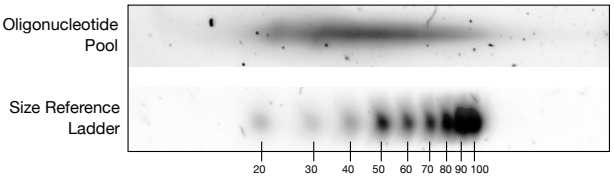


Figure 19. Gel electrophoresis results showing the distribution of strand lengths from the DNA synthesis process.

References

[1] L. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266(5187):1021–1024, 1994.

[2] M. E. Allentoft, M. Collins, D. Harker, J. Haile, C. L. Oskam, M. L. Hale, P. F. Campos, J. A. Samaniego, M. T. P. Gilbert, E. Willerslev, G. Zhang, R. P. Scofield, R. N. Holdaway, and M. Bunce. The half-life of DNA in bone: measuring decay kinetics in 158 dated fossils. *Proceedings of the Royal Society of London B: Biological Sciences*, 279(1748):4724–4733, 2012.

[3] C. Bancroft, T. Bowler, B. Bloom, and C. T. Clelland. Long-term storage of information in DNA. *Science*, 293(5536): 1763–1765, 2001.

[4] R. Carlson. Time for new DNA synthesis and sequencing cost curves. `http://www.synthesis.cc/2014/02/time-for-new-cost-curves-2014.html`, 2014.

[5] Y.-J. Chen, N. Dalchau, N. Srinivas, A. Phillips, L. Cardelli, D. Soloveichik, and G. Seelig. Programmable chemical controllers made from DNA. *Nature Nanotechnology*, 8(10):755–762, 2013.

[6] G. M. Church, Y. Gao, and S. Kosuri. Next-generation digital information storage in DNA. *Science*, 337(6102):1628, 2012.

[7] C. T. Clelland, V. Risca, and C. Bancroft. Hiding messages in DNA microdots. *Nature*, 399:533–534, 1999.

[8] ExtremeTech. New optical laser can increase DVD storage up to one petabyte. `http://www.extremetech.com/computing/159245-new-optical-laser-can-increase-dvd-storage-up-to-one-petabyte`, 2013.

[9] D. G. Gibson, J. I. Glass, C. Lartigue, V. N. Noskov, R.-Y. Chuang, M. A. Algire, G. A. Benders, M. G. Montague, L. Ma, M. M. Moodie, C. Merryman, S. Vashee, R. Krishnakumar, N. Assad-Garcia, C. Andrews-Pfannkoch, E. A. Denisova, L. Young, Z.-Q. Qi, T. H. Segall-Shapiro, C. H. Calvey, P. P. Parmar, C. A. Hutchison, H. O. Smith, and J. C. Venter. Creation of a bacterial cell controlled by a chemically synthesized genome. *Science*, 329(5987):52–56, 2010.

[10] N. Goldman, P. Bertone, S. Chen, C. Dessimoz, E. M. LeProust, B. Sipos, and E. Birney. Towards practical, high-capacity, low-maintenance information storage in synthesized DNA. *Nature*, 494:77–80, 2013.

[11] 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. *Angew. Chem. Int. Ed.*, 54:2552–2555, 2015.

[12] Q. Guo, K. Strauss, L. Ceze, and H. Malvar. High-density image storage using approximate memory cells. In *ASPLOS*, 2016.

[13] D. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[14] IDC. Where in the world is storage. `http://www.idc.com/downloads/where_is_storage_infographic_243338.pdf`, 2013.

[15] S. Kosuri and G. M. Church. Large-scale de novo DNA synthesis: technologies and applications. *Nature Methods*, 11:499–507, 2014.

[16] A. Leier, C. Richter, W. Banzhaf, and H. Rauhe. Cryptography with DNA binary strands. *Biosystems*, 57(1):13–22, 2000.

[17] M. D. Matteucci and M. H. Caruthers. Synthesis of deoxyoligonucleotides on a polymer support. *Journal of the American Chemical Society*, 103(11):3185–3191, 1981.

[18] R. Miller. Facebook builds exabyte data centers for cold storage. `http://www.datacenterknowledge.com/archives/2013/01/18/facebook-builds-new-data-centers-for-cold-storage/`, 2013.

[19] R. A. Muscat, K. Strauss, L. Ceze, and G. Seelig. DNA-based molecular architecture with spatially localized components. In *International Symposium on Computer Architecture*, 2013.

[20] T. P. Niedringhaus, D. Milanova, M. B. Kerby, M. P. Snyder, and A. E. Barron. Landscape of next-generation sequencing technologies. *Anal. Chem.*, 83:4327–4341, 2011.

[21] L. Qian, E. Winfree, and J. Bruck. Neural network computation with DNA strand displacement cascades. *Science*, 475(7356): 368–372, 2011.

[22] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[23] A. Sampson, J. Nelson, K. Strauss, and L. Ceze. Approximate storage in solid-state memories. In *International Symposium on Microarchitecture*, 2013.

[24] J. J. Schwartz, C. Lee, and J. Shendure. Accurate gene synthesis with tag-directed retrieval of sequence-verified DNA molecules. *Nature Methods*, 9(9):913–915, 2012.

[25] Sony. Sony develops magnetic tape technology with the world's highest recording density. `http://www.sony.net/SonyInfo/News/Press/201404/14-044E/`, 2014.

[26] K. Takahashi, S. Yaegashi, A. Kameda, and M. Hagiya. Chain reaction systems based on loop dissociation of DNA. In *DNA Computing*, volume 3892 of *Lecture Notes in Computer Science*, pages 347–358. Springer Berlin Heidelberg, 2006.

[27] B. Talawar. A crossbar interconnection network in DNA. In *Workshop on High Performance Computational Biology*, 2015.

[28] S. M. H. T. Yazdi, Y. Yuan, J. Ma, H. Zhao, and O. Milenkovic. A Rewritable, Random-Access DNA-Based Storage System. *Nature Scientific Reports*, 5(14318), 2015.

[29] J. N. Zadeh, B. R. Wolfe, and N. A. Pierce. Nucleic acid sequence design via efficient ensemble defect optimization. *Journal of Computational Chemistry*, 32(3):439–452, 2011.