DNA-Based Cryptography *

Ashish Gehani, Thomas LaBean, and John Reif
Department of Computer Science, Duke University †

Abstract

Recent research has considered DNA as a medium for ultra-scale computation and for ultra-compact information storage. One potential key application is DNA-based, molecular cryptography systems. We present some procedures for DNA-based cryptography based on one-time-pads that are in principle unbreakable. Practical applications of cryptographic systems based on one-time-pads are limited in conventional electronic media by the size of the one-time-pad; however DNA provides a much more compact storage medium, and an extremely small amount of DNA suffices even for huge one-time-pads. We detail procedures for two DNA one-time-pad encryption schemes: (i) a substitution method using libraries of distinct pads, each of which defines a specific, randomly generated, pair-wise mapping; and (ii) an XOR scheme utilizing molecular computation and indexed, random key strings. These methods can be applied either for the encryption of natural DNA or for artificial DNA encoding binary data. In the latter case, we also present a novel use of chip-based DNA micro-array technology for 2D data input and output. Finally, we examine a class of DNA steganography systems, which secretly tag the input DNA and then hide it within collections of other DNA. We consider potential limitations of these steganographic techniques, proving that in theory the message hidden with such a method can be recovered by an adversary. We also discuss various modified DNA steganography methods which appear to have improved security.

1 Introduction

1.1 Biomolecular Computation

Recombinant DNA techniques have been developed for a wide class of operations on DNA and RNA strands. There has recently arisen a new area of research known as DNA computing, which makes use of recombinant DNA techniques for doing computation, surveyed in [Rei98b]. Recombinant DNA operations were shown to be theoretically sufficient for universal computation [Hea92]. Biomolecular computing (BMC) methods have been proposed to solve difficult combinatorial search problems such as the Hamiltonian path problem [Adl94], using the vast parallelism available to do the combinatorial search among a large number of possible solutions represented by DNA strands. For example, [BDL96] and [RWB$^+$96] propose BMC methods for breaking the Data Encryption Standard (DES). While these methods for solving hard combinatorial search problems may succeed for fixed sized problems, they are ultimately limited by their volume requirements, which may grow exponentially with input size. However, BMC has many exciting further applications beyond pure combinatorial search. For example, DNA and RNA are appealing media for data storage due to the very large amounts of data that can be stored in compact volume. They vastly exceed the storage capacities of conventional electronic, magnetic, optical media. A gram of DNA contains about 10^{21} DNA bases, or about 10^8 tera-bytes. Hence, a few grams of DNA may have the potential of storing all the data stored in the world. Engineered DNA might be useful as a

*Supported by Grants NSF/DARPA CCR-9725021, CCR-96-33567, NSF IRI-9619647, ARO contract DAAH-04-96-1-0448, and ONR contract N00014-99-1-0406. A preliminary version appears in DIMACS DNA Based Computers V, American Mathematical Society, 2000.

†Department of Computer Science, Duke University, Box 90129, Durham, NC 27708-0129. E-mail: {geha,thl,reif}@cs.duke.edu

database medium for storing at least two broad classes of data: (i) processed, biological sequences, and (ii) conventional data from binary, electronic sources. Baum [Bau95] has discussed methods for fast associative searches within DNA databases using hybridization. Other BMC techniques [Rei98a] might perform more sophisticated database operations on DNA data such as database join operations and various massively parallel operations on the DNA data.

1.2 Cryptography

Data security and cryptography are critical aspects of conventional computing and may also be important to possible DNA database applications. Here we provide basic terminology used in cryptography [Sch96]. The goal is to transmit a message between a sender and receiver such that an eavesdropper is unable to understand it. Plaintext refers to a sequence of characters drawn from a finite alphabet, such as that of a natural language. Encryption is the process of scrambling the plaintext using a known algorithm and a secret key. The output is a sequence of characters known as the ciphertext. Decryption is the reverse process, which transforms the encrypted message back to the original form using a key. The goal of encryption is to prevent decryption by an adversary who does not know the secret key. An unbreakable cryptosystem is one for which successful cryptanalysis is not possible. Such a system is the one-time-pad cipher. It gets its name from the fact that the sender and receiver each possess identical notepads filled with random data. Each piece of data is used once to encrypt a message by the sender and to decrypt it by the receiver, after which it is destroyed.

1.3 Our Results

This paper investigates a variety of biomolecular methods for encrypting and decrypting data that is stored as DNA. In Section 2, we present a class of DNA cryptography techniques that are in principle unbreakable. We propose the secret assembly of a library of one-time-pads in the form of DNA strands, followed by a number of methods to use such one-time-pads to encrypt large numbers of short message sequences. The use of such encryption with conventional electronic media is limited by the large amount of one-time-pad data which must be created and transmitted securely. Since DNA can store a significant amount of information in a limited physical volume, the use of DNA could mitigate this concern. In Section 3, we present an interesting concrete example of a DNA cryptosystem in which a two-dimensional image input is encrypted as a solution of DNA strands. We detail how these strands are then decrypted using fluorescent DNA-on-a-chip technology. Section 4 discusses steganographic techniques in which the DNA encoding of the plaintext is hidden among other unrelated strands rather than actually being encrypted. We analyze a recently published genomic steganographic technique [TRB99], where DNA plaintext strands were appended with secret keys and then hidden among many other irrelevant strands. While the described system is appealing for its simplicity, our entropy based analysis allows extraction of the message without knowledge of the secret key. We then propose improvements that increase the security of DNA steganography.

2 DNA Cryptosystems Using Random One-Time-Pads

One-time-pad encryption uses a codebook of random data to convert plaintext to ciphertext. Since the codebook serves as the key, if it were predictable (i.e. not random), then an adversary could guess the algorithm that generates the codebook, allowing decryption of the message. No piece of data from the codebook should ever be used more than once. If it was, then it would leak information about the probability distribution of the plaintext, increasing the efficiency of an attempt to guess the message. These two principles, true randomness and single use of pads, dictate certain features of the DNA sequences and of sequence libraries, which will be discussed further below. This class of cryptosystems using a secret random one-time-pad are the only cryptosystems known to be absolutely unbreakable [Sch96].

We will first assemble a large one-time-pad in the form of a DNA strand, which is randomly assembled from short oligonucleotide sequences, then isolated and cloned. These one-time-pads will be assumed to be constructed in secret, and we further assume that specific one-time-pads are shared in advance by both the sender and receiver of the secret message. This assumption requires initial communication of the one-time-pad between sender and receiver, which is facilitated by the compact nature of DNA.

We propose two methods whereby a large number of short message sequences can be encrypted: (i) the use of substitution, where we encrypt each message sequence using an associatively matched piece from the DNA pad; or (ii) the use of bit-wise XOR computation using a biomolecular computing technique. The decryption is done by similar methods.

It is imperative that the DNA ciphertext is not contaminated with any of the plaintext. In order for this be effected, the languages used to represent each should be mutually exclusive. The simplest way to create mutually exclusive languages is to use disjoint plain and ciphertext alphabets. This would facilitate the physical separation of plaintext strands from the ciphertext using a set of complementary probes. If the ciphertext remains contaminated with residual plaintext strands, further purification steps can be utilized, such as the use of the DNA-SIEVE described in Section 4.4.

2.1 DNA Cryptosystem Using Substitution

A substitution one-time-pad system uses a plaintext binary message and a table defining a random mapping to ciphertext. The input strand is of length n and is partitioned into plaintext words of fixed length. The table maps all possible plaintext strings of a fixed length to corresponding ciphertext strings, such that there is a unique reverse mapping.

Encryption occurs by substituting each plaintext DNA word with a corresponding DNA cipher word. The mapping is implemented using a long DNA pad that consisting of many segments, each of which specifies a single plaintext word to cipher word mapping. The plaintext word acts as a hybridization site for the binding of a primer, which is then elongated. This results in the formation of a plaintext-ciphertext word-pair. Further, cleavage of the word-pairs and removal of the plaintext portion must occur. A potential application is detailed in Section 3.

An ideal one-time-pad library would contain a huge number of pads and each would provide a perfectly unique, random mapping from plaintext words to cipher words. Our construction procedure approaches these goals. The structure of an example pad is given in Figure 1.

The repeating unit is made up of: (i) one sequence word, C_i , from the set of cipher or codebook-matching words; (ii) one sequence word, P_i , from the set of plaintext words; and (iii) a polymerase "stopper" sequence. We note that each P_i includes a unique subsequence, which prevents frequency analysis attacks by mapping multiple instances of the same message plaintext to different ciphertext words. Further, this prefix could optionally be used to encode the position of the word in the message.

Each sequence pair i, uniquely associates a plaintext word with a cipher word. Oligo with sequence $\overline{P_i}$, corresponding to the Watson-Crick complement of plaintext word P_i , can be used as polymerase primer and be extended by specific attachment of the complement of cipher word C_i.

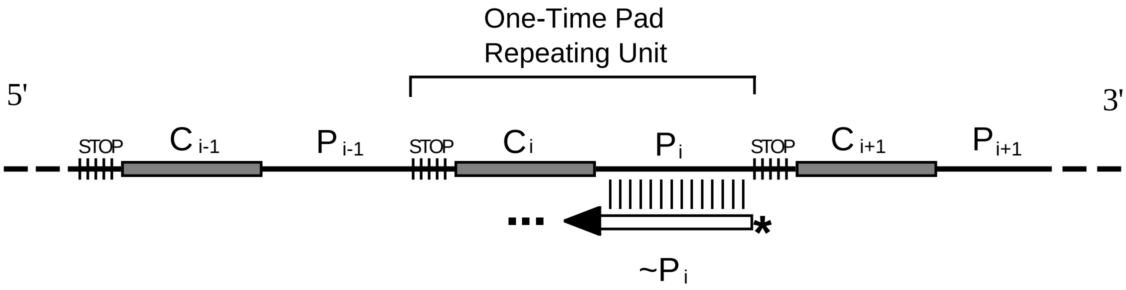

Figure 1: One-time-pad Codebook DNA Sequences

The stopper sequence prohibits extension of the growing DNA strand beyond the boundary of the paired cipher word. A library of unique codebook strands is constructed using this theme. Each individual strand from this codebook library specifies a particular, unique set of word pairings.

The one-time-pad consists of a DNA strand of length n containing $d = \frac{n}{L_1+L_2+L_3}$ copies of the repeating pattern: a cipher word of length L_2, a plaintext word of length L_1, and stopper sequence of length L_3. We note that word length grows logarithmically in the total pad length; specifically $L_1 = c_1 \log_2 n, L_2 = c_2 \log_2 n$, and $L_3 = c_3$, for fixed integer constants $c_1, c_2, c_3 > 1$. Each repeat unit specifies a single mapping pair, and no codebook word or plaintext word will be used more than once on any pad. Therefore, given a cipher word C_i we are assured that it maps to only a single plaintext word P_i and vice versa. The stopper sequence acts as "punctuation" between repeat units so that DNA polymerase will not be able to continue copying down the template (pad) strand. Stopper sequences consist of a run of identical nucleotides which act to halt strand copying by DNA polymerase given a lack of complementary nucleotide triphosphate in the test tube. For example, the sequence TTTT will act as a stop point if the polymerization mixture lacks its base-pairing complement, A. Stopper sequences of this variety have been prototyped previously [HAK$^+$97]. Given this structure, we can anneal primers and extend with polymerase in order to generate a set of oligonucleotides corresponding to the plaintext/cipher lexical pairings.

The experimental feasibility depends upon the following factors: (i) the size of the lexicon, which is the number of plaintext-ciphertext word-pairs, (ii) the size of each word, (iii) the number of DNA one-time-pads that can be constructed in a synthesis cycle, and (iv) the length of each message that is to be encrypted. If the lexicon used consisted of words of the English language, its size would be in the range of 10,000 to 25,000 word-pairs. If for experimental reasons, a smaller lexicon is required, then the words used could represent a more basic set such as ASCII characters, resulting in a lexicon size of 128. The implicit tradeoff is that this would increase message length. We estimate that in a single cloning procedure, we can produce 10^6 to 10^8 different one-time-pad DNA sequences. Choice of word encodings must guarantee an acceptable Hamming distance between sequences such that the fidelity of annealing is maximized. When generating sequences that will represent words, the space of all possibilities is much larger than the set that is actually needed for the implementation of the words in the lexicon. We also note that if the lexicon is to be split among multiple DNA one-time-pads, then care should be taken during pad construction to prevent a single word from being mapped to multiple targets.

If long-PCR with high fidelity enzymes introduces errors and the data in question is from an electronic source, we can pre-process it using suitable error-correction coding. If instead we are dealing with a wet database, the DNA one-time-pad's size can be restricted. This is done by splitting the single long one-time-pad into multiple shorter one-time-pads. In this case each cipher word would be modified to include a subsequence prefix that would denote which shorter one-time-pad should be used for its decryption. This increases the difficulty of cloning the entire set of pads.

The entire construction process can be repeated to prepare greater numbers of unique pads. Construction of the libraries of codebook pads can be approached using segmental assembly procedures used successfully in previous gene library construction projects [LK93, LB97] and DNA word encoding methods used in DNA computation [DMG$^+$96, DMG$^+$98, RDG$^+$97, FTC$^+$97, GFB$^+$96a, Mir96]. One methodology is chemical synthesis of a diverse segment set followed by random assembly of pads in solution. An issue to consider with this methodology is the difficulty of achieving full coverage while avoiding possible conflicts due to repetition of plaintext or cipher words. We can set the constants c_1 and c_2 large enough so that the probability of getting repeated words on a pad of length n is acceptably small.

Another methodology would be to use a commercially available DNA chip [FRP$^+$91, PSS$^+$94, CYH$^+$96, BKH96, Suy98]. See [JYP98] for previous use of DNA chips for I/O. The DNA chip has an array of immobilized DNA strands, so that multiple copies of a single sequence are grouped together in a microscopic pixel. The microscopic arrays of the DNA chip are optically addressable, and there is known technology for growing distinct DNA sequences at each optically addressable site of the array. Light-directed synthesis allows the chemistry of DNA synthesis to be conducted in parallel at thousands of locations, i.e. it is a combinatorial synthesis. Therefore, the number of sequences prepared far exceeds the number of chemical reactions required. For preparation of oligonucleotides of length L, the 4^L sequences are synthesized in $4L$ chemical reactions. For example, the $\sim 65,000$ sequences of length 8 require 32 synthesis cycles, and the 1.67×10^7 sequences of length 12 require only 48 cycles. Each pixel location on the chip comprises 10 microns square, so the complete array of 12-mer sequences could be contained within a ~ 4 cm square.

2.2 DNA XOR One-time-pad Cryptosystem

The Vernam cipher [Kah67] uses a sequence, S, of R uniformly distributed random bits known as a one-time-pad. A copy of S is stored at both the sender's and receiver's locations. L is the number of bits of S that have not yet been used for encrypting a message. Initially $L = R$. XOR operates on two binary inputs, yielding 0 if they are the same and 1 otherwise. When a plaintext binary message M which is $n < L$ bits long needs to be sent, each bit M_i is XOR'ed with the bit $K_i = S_{R-L+i}$ to produce the encrypted bit $C_i = M_i \oplus K_i$ for $i = 1, \ldots, n$. The n bits of S that have been consumed are then destroyed at the source and the encrypted sequence $C = (C_1, C_2, \ldots, C_n)$ is dispatched to the destination. At the destination the identical process is repeated - that is the sequence C is used in the place of M, performing bitwise XOR with bits from S, destroying the bits of S after they are consumed. The self-inverse property of binary XOR results in the initial message being reproduced since $C_i \oplus K_i = M_i$ since $C_i = M_i \oplus K_i$ and $M_i \oplus K_i \oplus K_i = M_i$.

To implement this algorithm with DNA, we need methods to (i) encode a plaintext message, (ii) create a DNA one-time-pad and (iii) effect bitwise XOR in DNA. Several methods exist to effect binary addition and XOR with DNA. In 1996, [GFB96b] prototyped single bit addition. Subsequent proposals [OGB97, GPZ97] allowed for chaining outputs with inputs, and parallel operations. [KLR98] experimentally demonstrated a logically reversible conditional XOR that required $O(n)$ recombinant DNA operations to act on n bit data. [LWR99] described a specific DNA tiling implementation of XOR and addition, based on previous work on self-assembly of DNA tilings [Win95a, Win95b, Win98, WLWS98, WYS96, Rei97]. An example of cumulative XOR using self-assembled DNA tilings has recently been published [MLRS00].

DNA tiles are multi-strand complexes containing two or more double helical domains such that individual oligonucleotide chains might base-pair in one helix then cross-over and base-pair in

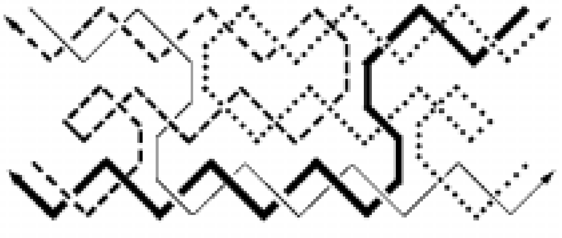

Figure 2: TAO triple-helix tile

another helix. Complexes involving crossovers (strand exchange points) create tiles which are multivalent and can have quite high thermal stability. Many helix arrangements and strand topologies are possible and several have been experimentally tested [LYQS96, LYK$^+$00]. Tiles with specific uncomplemented sticky ends at the corners were constructed, with the purpose of effecting self-assembly.

A binary input string can be encoded using a single tile for each bit. The tiles are designed such that they assemble linearly to represent the binary string. The use of special connector tiles allow two such linear tile assemblies representing two binary input strings respectively, to come together and create a closed framework within which specially designed output tiles can fit. This process allows for unmediated parallel binary addition or XOR. At the end of the process, there exists a single strand that runs through the entire assembly which will contain the two inputs and the output, as a result of the special design of these tiles [LYK$^+$00, LWR99]. By using this property, we are able to effect the Vernam cipher in DNA.

In particular, we use TAO triple-helix tiles (see Figure 2). The tile is formed by the complementary hybridization of four oligonucleotide strands (shown as different line types with arrowheads on their 3' ends). The three double-helical domains are shown with horizontal helix axes where the upper and lower helices end with bare strand ends and the central helix is capped at both ends with hairpins. Paired vertical lines represent crossover points where two strands exchange between two helices. Uncomplemented sticky ends can be appended to the four corners of the tile and encode neighbour rules for assembly of larger structures including computational complexes. For more details see [LYK$^+$00, MLRS00].

We outline below how the bit-wise XOR operation may be done (see Figure 3). For each bit M_i of the message, we construct a sequence a_i that will represent the bit in a longer input sequence. By using suitable linking sequences, we can assemble the message M's n bits into the sequence $a_1 a_2 \ldots a_n$, which serves as the scaffold strand for one binary input to the XOR. The further portion of the scaffold strand $a'_1 a'_2 \ldots a'_n$ is created based on random inputs and serves as the one-time-pad. It is assumed that many scaffolds of the form $a'_1 a'_2 \ldots a'_n$ have been initially created, cloned using PCR [Bar94, Rob94] or an appropriate technique, and then separated and stored at both the source and destination points in advance. When encryption needs to occur at the source, the particular scaffold used is noted and communicated using a prefix index tag that both sender and destination know corresponds to a particular scaffold.

By introducing the scaffold for the message, the scaffold for the one-time-pad, connector tiles and the various sequences needed to complete the tiles, the input portion of the structure shown in Figure 3 forms. We call this the input assembly. This process of creating input scaffolds and assembling tiles on the scaffold has been carried out successfully [LWR99]. Each pair of tiles (corresponding to a pair of binary inputs) in the input assembly creates a slot for the binding of a single output tile. When output tiles are introduced, they bind into appropriate binding slots by

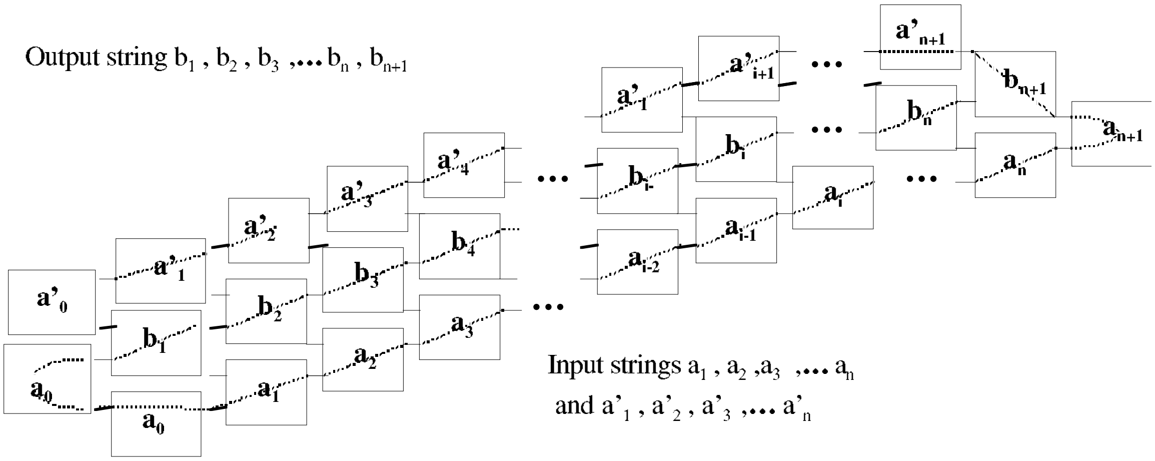
Output string b1 , b2 , b3 ,... bn , bn+1
a'0
a0
a0
a'1
b1
a1
a'2
b2
a2
a'3
b3
a'4
b4
a3
a'1
bi-
ai-2
a'i+1
bi
ai-1
ai
a'n+1
bn
an
bn+1
an+1
Input strings a1 , a2 ,a3 ,... an
and a'1 , a'2 , a'3 ,... a'n

Figure 3: XOR computation by the use of DNA tiles

the matching of sticky ends.

Finally, adding ligase enzyme results in a continuous reporter strand R that runs through the entire assembly. If $b_i = a_i \oplus a'_i$, for $i = 1, \ldots, n$, then the reporter $R = a_1 a_2 \ldots a_n . a'_1 a'_2 \ldots a'_n . b_1 b_2 \ldots b_n$. The reporter strand is shown as a dotted line in Figure 3. This strand may be extracted by first melting apart the hydrogen bonding between strands and then purifying by polyacrylamide gel electrophoresis. It contains the input message, the encryption key, and the ciphertext all linearly concatenated. The ciphertext can be excised using a restriction endonuclease if a cleavage site is encoded between the a_0 and b_1 tiles. Alternatively the reporter strand could incorporate a nick at that point by using an unphosphorylated oligo between those tiles. The ciphertext could then be gel purified since its length would be half that of the remaining sequence. This may then be stored in a compact form and sent to a destination.

Since XOR is its own inverse, the decryption of a Vernam cipher is effected by applying the identical process as encryption with the same key. Specifically, the $b_1 b_2 \ldots b_n$ is used as one input scaffold, the other is chosen from the stored $a'_1 a'_2 \ldots a'_n$ according to the index indicating which sequence was used as the encryption key. The sequences for tile reconstitution, the connector tiles, and ligase are added. After self-assembly, the reporter strand is excised, purified, cut at the marker and the plain text is extracted.

We need to guard against loss of synchronization between the message and the key, which would occur when a bit is spuriously introduced or deleted from either sequence. Some fault tolerance is provided by the use of several nucleotides to represent each bit in the the tiles' construction. This reduces the probabiliity of such errors.

3 Encrypting Images with DNA Chips and DNA One-Time-Pads

3.1 Overview of Method

In this section we outline a system capable of encryption and decryption of input and output data in the form of 2D images recorded using microscopic arrays of DNA on a chip. The system we describe here consists of: a data set to be encrypted, a chip bearing immobilized DNA strands, and a library of one-time-pads encoded on long DNA strands as described in Section 2.1. The data set for encryption in this specific example is a 2-dimensional image, but variations on the method may be useful for encoding and encrypting other forms of data or types of information. The DNA

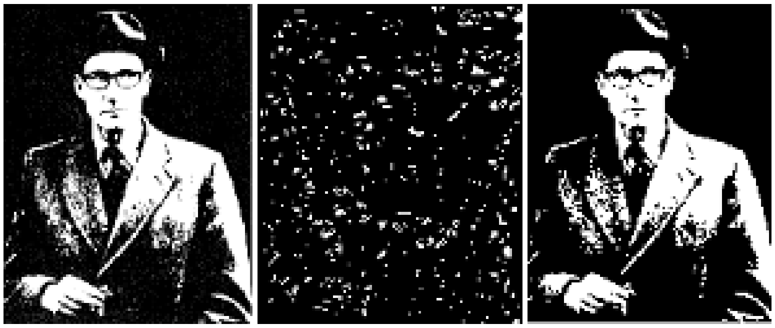

Figure 4: DNA Chip Input/Output: Panel A: Message, Panel B: Encrypted Message, Panel C: Decrypted Message

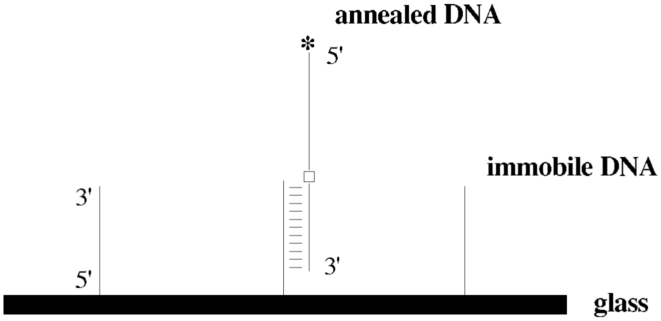

Figure 5: Components and Organization of the DNA Chip

chip contains an addressable array of nucleotide sequences immobilized such that multiple copies of a single sequence are grouped together in a microscopic pixel. Such DNA chips are currently commercially available and chemical methods for construction of custom variants are well developed. Further chip details will be given below.

Figure 4 gives a coarse grained outline of the I/O method. Fluorescently labeled, word-pair DNA strands are prepared from a substitution pad codebook as described in Section 2.1. These are annealed to their sequence complements at unique sites (pixels) on the DNA chip. The message information (Panel A) is transferred to a photo mask with transparent (white) and opaque (black) regions. Following a light-flash of the mask-protected chip, the annealed oligonucleotides beneath the transparent regions are cleaved at a photo-labile position. Their $5'$ sections dissociate from the annealed $3'$ section and are collected in solution. This test tube of fluorescently labeled strands is the encrypted message. Annealed oligos beneath the opaque regions are unaffected by the light-flash and can be subsequently washed off the chip and discarded. If the encrypted message oligos are reannealed onto a different DNA chip, they would anneal to different locations and the message information would be unreadable (Panel B). Note that if one used a chip identical to the encoding chip, and if the sequence lexicons for $5'$ segment (cipher word) and $3'$ segment (plaintext word) are disjoint, no binding would occur and the chip in Panel B would be completely black. Decrypting is accomplished by using the fluorescently labeled oligos as primers in asymmetric PCR with the same one-time codebook which was used to prepare the initial word-pair oligos. When the word-pair PCR product is bound to the same DNA chip, the decrypted message is revealed (Panel C).

The annealed DNA in Figure 5 corresponds to the word-pair strands prepared from a random substitution pad as described in Section 2.1 above. Immobile DNA strands are located on the glass substrate of the chip in a sequence addressable grid according to currently used techniques.

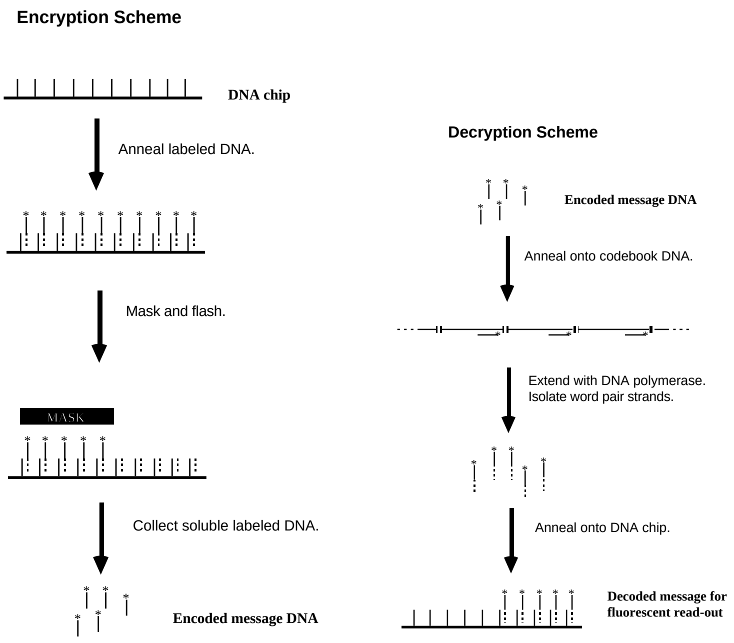

Figure 6: Step by step procedures for encryption and decryption

Ciphertext-plaintext word-pair strands anneal to the immobile ones. The annealed strand contains a fluorescent label on its $5'$ end (denoted with an asterisk in the figure). This is followed by the complement of a plaintext word (uncomplemented section) and the complement of a cipher word (complemented section). Located between the two words is a photo-cleavable base analog (white box in the figure) capable of cleaving the backbone of the oligonucleotide.

Figure 6 gives step by step procedures for encryption and decryption. For encryption, we start with a DNA chip displaying the sequences drawn from the cipher lexicon. In step one, the fluorescently labeled word-pair strands prepared from a one-time-pad are annealed to the chip at the pixel bearing the complement to their $3'$ end. In the next step, the mask (heavy black bar) protects some pixels from a light-flash. At unprotected regions, the DNA backbone is cleaved at a site between the plaintext and cipher words. In the final step, the $5'$ segments, still labeled with fluorophore at their $5'$ ends, are collected and transmitted as the encrypted message.

A message can be decrypted only by using the one-time-pad and DNA chip identical to those used in the encryption process. First, the word-pair strands must be reconstructed by appending the proper cipher word onto each plaintext word. This is accomplished by primer extension or asymmetric PCR using transmitted strands as primer and one-time-pad as template. The strands bind to their specific locations on the pad and are appended with their proper cipher partner. Note that in the decrypting process the fluorescent label is still required, but the photo-labile base is unnecessary and not present. The final step of decryption involves binding the reformed word-pair strands to the DNA chip and reading the message by fluorescent microscopy.

3.2 Additional Technical Considerations

Some details concerning the configuration of the DNA chip should be mentioned. In the current incarnation of the method, reverse synthesis of oligos directly on the chip or “spot attachment” would be required. Chemical reagents for reverse synthesis are commercially available although not as widely used as those for 3’-to-5’ methods. Spot attachment of oligos onto the chip results in decreased pixel density and increased work. However, recent chip improvements, including etching with hydrophobic gridlines, may alleviate this drawback.

One potential photo-cleavable base analog is $7-$nitroindole nucleoside. It has previously been shown to produce a chemical lesion in the effected DNA strand which causes backbone cleavage. Use of $7-$nitroindole nucleoside for strand cleavage has been shown to produce useable $5'$ and $3'$ ends [KBDL98]. Production of 'clean' $3'$ ends is critical for decrypting the message, since the cipher strands must hybridize to the one-time-pad and act as primers for the polymerase mediated strand elongation (PCR). Primers and templates containing the $7-$nitroindole nucleoside have been shown to function properly in PCR and other enzymatic reactions.

4 DNA Steganography Analysis

Steganography using DNA is appealling due to its simplicity. One method proposed involves taking "plaintext" input DNA strands, tagging each with "secret key" strands, and then hiding them among random "distracter" strands. The plaintext is retrieved by hybridization with the complement of the secret key strands. It has been postulated that in the absence of knowledge of the secret key, it would be necessary to examine all the strands including the distracters to retrieve the plaintext. Based on the likely difference in entropy of the distracters and the plaintext, we argue that the message can be retrieved without the key.

4.1 Relevant Data Compression Result

The *compression ratio* is the quotient of the length of the compressed data divided by the length of the source data. For example, many images may be losslessly compressed to a 1/4 of their original size; English text and computer programs have compression ratios of about 1/3; most DNA has a compression ratio between 1/2 and 1/1.2 [GT94, LY97]. Protein coding sequences make efficient use of amino acid coding and have larger compression ratios [NMW99, HH92]. The Shannon information theoretic *entropy rate* is denoted by $H_S \leq 1$. It is defined to be the rate that the entropy increases per symbol, for a sequence of length n with $n \to \infty$ [CT91]. It provides a measure of the asymptotic rate at which a source can be compressed without loss of information. Random sequences can not be compressed and therefore have an entropy rate of 1.

Lossless data compression with an algorithm such as Lempel-Ziv [ZL77], is theoretically asymptotically optimal. For sequences whose length n is large, the compression ratio approaches the entropy rate of the source. In particular, it is of the form $(1+\epsilon(n))H_S$, where $\epsilon(n) \to 0$ for $n \to \infty$. Algorithms such as Lempel-Ziv build an indexed dictionary of all subsequences parsed that can not be constructed as a catenation of current dictionary entries. Compression is performed by sequentially parsing the input text, finding maximal length subsequences already present in the dictionary, and outputting their index number instead. When a subsequence is not found in the dictionary, it is added to it (including the case of single symbols). Algorithms can achieve better compression by making assumptions about the distribution of the data [BWC89]. It is possible to use a dictionary of bounded size, consisting of the most frequent subsequences. Experimentation on a wide variety of text sources shows that this method can be used to achieve compression within a small percentage of the ideal [Sto88]. In particular, the compression ratio is of the form $(1+\epsilon)H_S$, for a small constant $\epsilon > 0$ typically of at most 1/10 if the source length is large.

Lemma 1 *The expected length of a parsed word is between* $\frac{L}{1+\epsilon}$ *and* L, *where* $L = \frac{\log_b n}{H_S}$.

Proof Assume the source data has an alphabet of size b. An alphabet of the same size can be used for the compressed data. The dictionary can be limited to a constant size. We can choose an algorithm that achieves a compression ratio within $1+\epsilon$ of the asymptotic optimal, for a small $\epsilon > 0$. Therefore, for large n, we can expect the compression ratio to approach $(1+\epsilon)H_S$.

Each parsed word is represented by an index into the dictionary, and so its size would be $\log_b n$ if the source had no redundancy. By the choice of compression algorithm, the length of the compressed data is between H_S and $H_S(1+\epsilon)$ times the length of the original data. From these two facts, it follows that the expected length of a code word will be between $\frac{\log_b n}{(1+\epsilon)H_S}$ and $\frac{\log_b n}{H_S}$. □

Lemma 2 *A parsed word has length* $\leq \frac{L}{p}$ *with probability* $\geq 1-p$.

Proof The probability of a parsed word having length $> \frac{L}{p}$ is $< p$, for all $p \in (0,1)$, by the Markov inequality. The lemma follows from this. □

Lemma 3 *A parsed word has length* $\geq c'L$ *with probability* $\geq 1-p$*, if* $p > 1 - \frac{1}{c(1+\epsilon)}$ *and* $c' = c - \frac{c-\frac{1}{1+\epsilon}}{p} > 0$.

Proof The maximum length of a parsed word has an upper bound in practice. We assume that this is cL for a small constant $c > 1$. We use Δ to denote the difference between the maximum possible and the actual length of a parsed word, and $\bar{\Delta}$ to denote the difference's expected value. Applying Lemma 1, $0 < \bar{\Delta} < cL - \frac{L}{1+\epsilon} (= (c - \frac{1}{1+\epsilon})L)$. The probability that $\Delta > \frac{\bar{\Delta}}{p} (= \frac{(c-\frac{1}{1+\epsilon})L}{p})$, is $< p$, by the Markov inequality. Therefore, with probability $< p$, a parsed word has length $< cL - \frac{\bar{\Delta}}{p} = c'L$, where $c' = c - \frac{c-\frac{1}{1+\epsilon}}{p}$. We choose $p > 1 - \frac{1}{c(1+\epsilon)}$ so that $0 < c' \leq c$, since parsed words must have positive length that does not exceed the maximum postulated. □

Lemma 4 *A parsed word has length between* $c'L$ *and* $\frac{L}{p}$ *with probability* $\geq (1-p)^2$*, if* $p > 1 - \frac{1}{c(1+\epsilon)}$ *and* $c' > H_S$.

Proof This follows from Lemmas 2 and 3. □

4.2 Analysis Assumptions

We assume all the following. The alphabet in question is that of DNA and therefore has size 4. The probability distribution of the "plaintext" DNA source S is known - for example, that it is generated by a stationary ergodic process. The "distracter" DNA strands have a random uniform distribution over the 4 DNA bases. Both "plaintext" and "distracter" DNA strands have the same length since they may otherwise be distinguished by length. A Lempel-Ziv algorithm variant that meets the criteria of Lemma 4 is known. p is fixed just above $1 - \frac{1}{c(1+\epsilon)}$. $f(n) \approx g(n)$ if $\frac{f(n)}{g(n)} \to 1$ and $(1-\frac{1}{n})^n \approx \frac{1}{e}$, for large n.

4.3 Constructing A Dictionary

Let $L = H_S \log_4 n$. D is the set of d most frequently occurring words of the source, where d is the size of the dictionary. D' is the subset of D that consists of words that meet the following two criteria. The first is that the word's length must be between $c'L$ and $\frac{L}{p}$. The second is that the word's frequency in the source S must be $> \frac{1}{n'}$, where $n' = (1-p)^2 \frac{n}{L}$.

Lemma 5 *The probability that a word* w *in* D' *is a parsed word of the "plaintext" DNA sequence is* $> 1 - \frac{1}{e}$.

Proof Let X be a "plaintext" DNA sequence of length n. Consider D'', the subset of D containing words of length between $c'L$ and $\frac{L}{p}$. D'' contains at least $(1-p)^2$ of the parsed words of X by Lemma 4. D' is the subset of D'' which consists of only words that have frequency $> \frac{1}{n'}$. Consider a word v parsed from X. The probability that a word w from D' is not v is $< 1 - \frac{1}{n'}$ by construction. X has an expected number $\frac{n}{L}$ parsed words. By Lemma 1, there are an expected number $(1-p)^2 \frac{n}{L}$ words with length in the range between $c'L$ and $\frac{L}{p}$. The probability that w is none of these words is therefore $< (1 - \frac{1}{n'})^{(1-p)^2 \frac{n}{L}} \approx \frac{1}{e}$. Thus, a word w in D' is some parsed word of X with probability $> 1 - \frac{1}{e}$. □

4.4 DNA-SIEVE : Plaintext Extraction Without The Key

Strand separation operates on a test tube of DNA, using recombinant DNA technology to split the contents into two tubes, T and T' with separation error rates $\sigma^-, \sigma^+ : 0 < \sigma^-, \sigma^+ < 1$. The goal is to transfer all the strands that contain the subsequence w into the tube T', leaving all the rest in tube T. A fraction $< \sigma^-$ of the strands without subsequence w enter T'. A fraction $> 1 - \sigma^+$ of the strands containing w are left in T. We assume that $\rho = \frac{\sigma^-}{(1-\sigma^+)(1-\frac{1}{e})}, 0 < \rho < 1$. Modest expectations for separation technology yield $0 < \sigma^+ < 0.2$ and $0 < \sigma^+ < 0.05$. Using $\sigma^- = \sigma^+ = 0.2$ suffices to obtain ρ in the desired range.

DNA-SIEVE is to be used to extract the "plaintext" DNA strands from the mix in which there are many "distracters". It begins with a tube T. The separate operation is iteratively applied. In each round, a previously unused word w from the set D' is chosen. All strands that contain it are retained by using hybridization with the complement of w. We use $r(T)$ to denote the ratio of the distracters to the plaintext, and $F(T)$ to denote the tube from which the strands with subsequence w were removed.

4.5 DNA-SIEVE Analysis

The success of DNA-SIEVE rests on the fact that a word in D' is likely to occur in plaintext X with probability $1 - \frac{1}{e}$, while it is expected to occur in a random text R with probability close to 0.

Lemma 6 *The probability that a word in D' is a subsequence of R is $\approx n4^{-c'L} = \frac{1}{n^{\frac{c'}{H_S}-1}}$.*

Proof Let R denote a random "distracter" sequence of length n over the alphabet of the 4 DNA bases. Since all sequences are equiprobable, one of length $c'L = c'\frac{\log_4 n}{H_S}$ is likely to occur with probability $4^{-c'L} = 4^{\log_4 n^{\frac{-c'}{H_S}}} = \frac{1}{n^{\frac{c'}{H_S}}}$. Since it can occur at any of $\approx n$ locations in R, the probability of it occurring in R is $n4^{-c'L} = \frac{1}{n^{\frac{c'}{H_S}-1}}$. By assumption in Lemma 4, $c' > H_S$, so $\frac{c'}{H_S} - 1 > 0$. □

Lemma 7 *If DNA-SIEVE operates on tube T and results in tube $F(T)$, then at the most $\approx \sigma^-$ of the distracters in T are in $F(T)$, while at least $\approx (1-\sigma^+)(1-\frac{1}{e})$ of the plaintext strands of T are in $F(T)$.*

Proof The probability that a distracter strand in T is present in $F(T)$ is limited by $\sigma^- + n4^{-c'L}$, the sum of the error and the theoretical chance. By assumption, the error rate is $< \sigma^-$. By the Lemma 6 the chance is $< \frac{1}{n^{\frac{c'}{H_S}-1}}$. Since n is large, this is ≈ 0. Therefore at most σ^- of the distracters in T reach $F(T)$. Similarly, by Lemma 5, at least $1 - \frac{1}{e}$ of the plaintext strands in T are expected to be in $F(T)$. By assumption, at most σ^+ of the strands that should reach $F(T)$ are left behind due to separation error. Therefore, $\approx (1 - \sigma^+)(1 - \frac{1}{e})$ of the plaintext strands actually reach $F(T)$. □

Lemma 8 *The probability distribution of the distracter strands returns to the original one after an expected number of* $\frac{2L}{p}$ *iterations of DNA-SIEVE.*

Proof Each iteration of DNA-SIEVE uses a word w from D' that has not been previously used. Assume w is a prefix or suffix of another word w' in D'. Once DNA-SIEVE has been using w, the probability distribution of the distracters complementary to w' will be altered. The distribution of the rest will remain unaffected. There are $\leq \frac{2L}{p}$ words that can overlap with a given word from D'. Therefore, a particular separation affects $\leq \frac{2L}{p}$ other iterations. If w is chosen randomly from D', then after an expected number of $\frac{2L}{p}$ iterations, all the strands will be equally effected and hence the probability distribution of the distracters will be the same as before the sequence of iterations. Such a number of iterations is termed "independent". □

Theorem 1 *To reduce the ratio of distracter to plaintext strands by a factor* r*, it suffices to apply DNA-SIEVE an expected number of* $O(\log n)\log r$ *times.*

Proof Denote the ratio of the distracter strands to the plaintext strands in test tube T with r'. Now consider a tube $F(T)$ that results from applying DNA-SIEVE t times till this ratio has been reduced by a factor r. Denote the ratio for tube $F(T)$ by r''. By Lemma 8, after an expected number of $\frac{2L}{p}$ iterations, a test tube $G(T)$ is produced with the same distribution of distracters as in T. Applying Lemma 7, we expect that after every set of $\frac{2L}{p}$ iterations, the ratio will change by at least $\rho = \frac{\sigma^-}{(1-\sigma^+)(1-\frac{1}{e})}$. We expect a decrease in the concentration after t iterations by a factor of $\rho^{\frac{t}{\frac{2L}{p}}}$. To attain a decrease of $\frac{r''}{r'}$, we need $t = \frac{2L}{p}\log\frac{r''}{r'}\log\rho$. Since $L = O(\log n)$ and $\rho = O(1)$, $t = O(\log n)\log r$. □

4.6 DNA-SIEVE Implementation Considerations

The theoretical analysis of DNA-SIEVE was used to justify the expected geometric decrease in the conentration of the distracter strands. It also provides two further insights. The number of "plaintext" DNA strands may decrease by a factor of $(1 - \sigma^+)(1 - 1/e)$ after each iteration. It is therefore prudent to increase the absolute number of copies periodically (by ligating all the strands with primer sequences, PCR cycling, and then digesting the primers that were added). The number of iterations that can be performed is limited to n' due to the fact that a distinct word from D' must be used each time. This limits the procedure to operation on a population where the number of distracter strands is $< 4^{n'}$.

4.7 Empirical Analysis

We performed an empirical analysis of DNA-SIEVE. We assumed that the test tube would start with 10^8 distracter strands and 10^3 message strands. The first parameter varied was the number of "indpendent" iterations of DNA-SIEVE - from 1 to 10. The second parameter varied was the separation error rate - from 0.01 to 0.25 in multiples of 0.01. Here we do not assume a difference in the error rate for false positives and false negatives that occur during the separation. In each case,

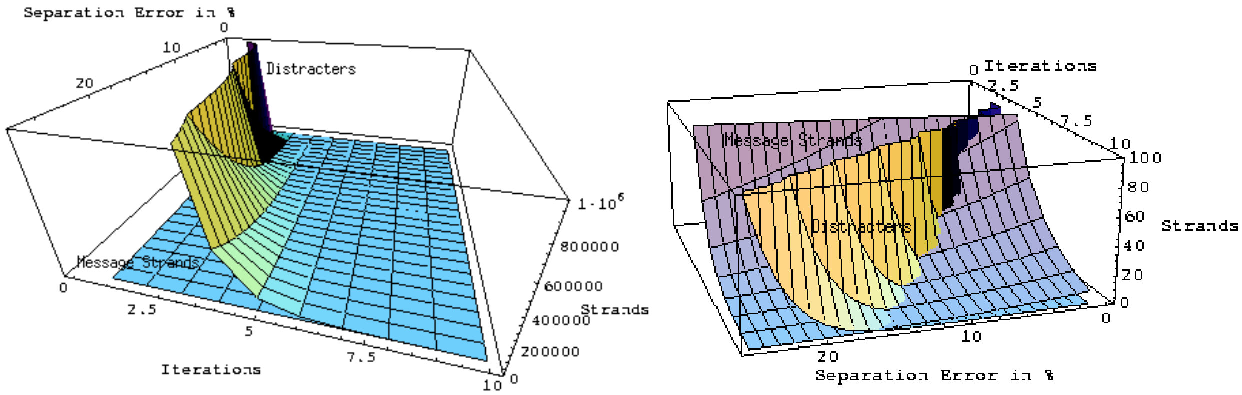

Figure 7: Simulation of DNA-SIEVE : Distracters have a sharper drop-off in concentration

the number of distracters and message strands remaining was computed. The results are plotted in Figure 7. From this we can see that 5 to 10 iterations of DNA-SIEVE suffice to reduce the distracter population's size to below that of the message strands when the separation error is < 0.18. The table illustrates the number of distracters and message strands left after 3, 6 and 9 iterations with varying separation error rates. If the error rate is reasonable, it can be seen from the table that there remain enough message strands for the plaintext to be found. If the separation error rate is high, the number of strands used must be increased to allow enough iterations of DNA-SIEVE to occur before the message strands are all lost.

Iterations	Separation Error	0.05	0.1	0.15	0.2	0.25
3	Distracters	12500	100000	337500	800000	1562500
	Messages	217	184	155	129	107
6	Distracters	2	100	1139	6400	24414
	Messages	47	34	24	17	11
9	Distracters	0	0	4	51	381
	Messages	10	6	4	2	1

4.8 Improving DNA Steganography

We can improve a DNA steganography system by reducing the difference between the plaintext and distracter strands. This can be done by making the distracters similar to the plaintext by creating them using random assembly of elements from the dictionary D. Alternatively, DNA-SIEVE can be employed on a set of random distracters to shape the population into one whose distribution matches that of the plaintext. We note, however, that if the relative entropy [CT91] between the plaintext and the distracter strand populations is large enough, DNA-SIEVE can be employed as previously described. An attacker can use a larger dictionary, which provides a better model of the plaintext source, to increase the relative entropy re-enabling the use of DNA-SIEVE. If the M plaintext strands are tagged with a sequence that allows them to be extracted, then they may be recognized by the high frequency of the tag sequence in the population. To guard against this, N sets of M strands each can be mixed in. This results in a system that uses $V = O(MN)$ volume. To prevent a brute force attack, N must be large, potentially detracting from the practicality of using using the DNA steganographic system.

The other approach to reduce the distinguishability of the plaintext from the distracters is to make the former mimic the latter. By compressing the plaintext with a lossless algorithm, such as Lempel-Ziv [ZL77], the relative entropy of the message and the distracter populations can be reduced. If the plaintext is derived from an electronic source, it can be compressed in a preprocessing step. If the source is natural DNA, it can be recoded using a substitution method similar to the one described in Section 2. However, the security of such a recoding remains unknown. In the case of natural DNA, for equivalent effort, DNA cryptography offers a more secure alternative to DNA steganography.

5 Conclusion

This paper presented an initial investigation into the use of DNA-based information security. We discussed two classes of methods: (i) DNA cyptography methods based on DNA one-time-pads, and (ii) DNA steganography methods. Our DNA substitution and XOR methods are based on one-time-pads, which are in principle unbreakable. We described our implementation of DNA cyptography with 2D input/output. We showed that a class of DNA steganography methods offer limited security and can be broken with a reasonable assumption about the entropy of the plaintext messages. We considered modified DNA steganographic systems with improved security. Steganographic techniques rest on the assumption that the adversary is unaware of the existence of the data. When this does not hold, DNA cryptography must be relied upon.

References

[Adl94] L. M. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024, November 1994.

[Bar94] W. M. Barnes. Pcr amplification of up to 35-kb dna with high fidelity and high yield from bacteriophage templates. *Proc. Natl. Acad. Sci.*, 91:2216–2220, 1994.

[Bau95] Eric B. Baum. Building an associative memory vastly larger than the brain. *Science*, 268:583–585, April 28, 1995.

[BDL96] Dan Boneh, Christopher Dunworth, and Richard J. Lipton. Breaking DES using a molecular computer. In Eric B. Baum and Richard J. Lipton, editors, *DNA Based Computers*, volume 27 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science.* American Mathematical Society, 1996.

[BKH96] A. P. Blanchard, R. J. Kaiser, and L. E. Hood. High-density oligonucleotide arrays. *Biosens. Bioelec.*, 11:687–690, 1996.

[BWC89] T. Bell, I. H. Witten, and J. G. Cleary. Modeling for text compression. *ACM Computing Surveys*, 21(4):557–592, December 1989.

[CT91] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* Wiley Series in Telecommunications. John Wiley & Sons, New York, NY, USA, 1991.

[CYH$^+$96] M. Chee, R. Yang, E. Hubbell, A. Berno, X. C. Huang, D. Stern, J. Winkler, D. J. Lockhart, M. S. Morris, and S. P. A. Fodor. Accessing genetic information with high-density DNA arrays. *Science*, 274:610–614, 1996.

[DMG$^+$96] R. Deaton, R. C. Murphy, M. Garzon, D. R. Franceschetti, and S. E. Stevens, Jr. Good encodings for DNA-based solutions to combinatorial problems. In *Proceedings of the Second Annual Meeting on DNA Based Computers*, DIMACS: Series in Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, May 1996.

[DMG+98] R. Deaton, R.C. Murphy, M. Garzon, D.R. Franceschetti, and Jr. S.E. Stevens. Reliability and efficiency of a dna-based computation. *Phys. Rev. Lett. 80*, pages 417–420, 1998.

[FRP+91] S. Fodor, J. L. Read, M. C. Pirrung, L. Stryer, A. Tsai Lu, and D. Solas. Light-directed spatially addressable parallel chemical synthesis. *Science*, 251:767–773, 1991.

[FTC+97] Anthony G. Frutos, Andrew J. Thiel, Anne E. Condon, Lloyd M. Smith, and Robert M. Corn. DNA computing at surfaces : 4 base mismatch word design. In David Wood, editor, *Proceedings of the 3rd DIMACS Workshop on DNA Based Computers, held at the University of Pennsylvania, June 23 – 25, 1997*, DIMACS: Series in Discrete Mathematics and Theoretical Computer Science., page 238, Providence, RI, June 1997. American Mathematical Society.

[GFB+96a] Jesse M. Gray, Tony G. Frutos, A. Michael Berman, Anne E. Condon, Max G. Lagally, Lloyd M. Smith, and Robert M. Corn. Reducing errors in DNA computing by appropriate word design, November 1996.

[GFB96b] Frank Guarnieri, Makiko Fliss, and Carter Bancroft. Making DNA add. *Science*, 273(5272):220–223, jul 1996.

[GPZ97] Vineet Gupta, Srinivasan Parthasarathy, and Mohammed J. Zaki. Arithmetic and logic operations with DNA. In David Wood, editor, *Proceedings of the 3rd DIMACS Workshop on DNA Based Computers, held at the University of Pennsylvania, June 23 – 25, 1997*, volume 48 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science.*, pages 212–220, Providence, RI, June 1997. American Mathematical Society.

[GT94] S. Grumbach and F. Tahi. A new challenge for compression algorithms: genetic sequences. *Inf. Proc. and Management*, 30(6):875–886, 1994.

[HAK+97] Masami Hagiya, Masanori Arita, Daisuke Kiga, Kansaku Sakamoto, and Shigeyuki Yokoyama. Towards parallel evaluation and learning of boolean μ-formulas with molecules. In David Wood, editor, *Proceedings of the 3rd DIMACS Workshop on DNA Based Computers, held at the University of Pennsylvania, June 23 – 25, 1997*, volume 48 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science.*, pages 105–114, Providence, RI, June 1997. American Mathematical Society.

[Hea92] Thomas Head. Splicing schemes and DNA. *Lindenmayer Systems; Impact on Theoretical computer science and developmental biology*, pages 371–383, 1992.

[HH92] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci*, 89(22):10915–9, Nov 1992.

[JYP98] Allen P. Mills Jr., Bernard Yurke, and Philip M. Platzman. Article for analog vector algebra computation. In Lila Kari, Harvey Rubin, and David Harlan Wood, editors, *Proceedings 4th DIMACS Workshop on DNA Based Computers, held at the University of Pennysylvania, Philadelphia, USA June 15 - June 19, 1998*, pages 175–180, 1998.

[Kah67] David Kahn. *The Codebreakers.* Macmillan, NY, 1967.

[KBDL98] M. Kotera, A. G. Bourdat, E. Defrancq, and J. Lhomme. A highly efficient synthesis of oligodeoxyribonucleotides containing the 2'-deoxyribonolactone lesion. *J. Am. Chem. Soc.*, 120:11810–1, 1998.

[KLR98] Joshua P. Klein, Thomas H. Leete, and Harvey Rubin. A biomolecular implementation of logical reversible computation with minimal energy dissipation. In Lila Kari, Harvey Rubin, and David Harlan Wood, editors, *Proceedings 4th DIMACS Workshop on DNA Based Computers, held at the University of Pennysylvania, Philadelphia, USA June 15 - June 19, 1998*, pages 15–23, 1998.

[LB97] T. H. LaBean and T.R. Butt. Methods and materials for producing gene libraries. U.S. Patent Number 5,656,467, August 1997.

[LK93] Thomas LaBean and S.A. Kauffman. Design of synthetic gene libraries encoding random sequence proteins with desired ensemble characteristics. *Protein Science*, 2:1249–1254, 1993.

[LWR99] Thomas H. LaBean, Erik Winfree, and John H. Reif. Experimental progress in computation by self-assembly of DNA tilings. In *DNA Based Computers V*, 1999. Article accepted to the DIMACS 5th International Meeting on DNA Based Computers.

[LY97] Loewenstern and Yianilos. Significantly lower entropy estimates for natural DNA sequences. In *DCC: Data Compression Conference*, pages 151–161. IEEE Computer Society TCC, 1997.

[LYK$^+$00] T. H. LaBean, H. Yan, J. Kopatsch, F. Liu, E. Winfree, H. J. Reif, and N. C. Seeman. The construction, analysis, ligation and self-assembly of DNA triple crossover complexes. *J. Am. Chem. Soc.*, 122:1848–1860, 2000.

[LYQS96] Xiaojun Li, Xiaoping Yang, Jing Qi, and Nadrian C. Seeman. Antiparallel DNA double crossover molecules as components for nanoconstruction. *Journal of the American Chemical Society*, 118(26):6131–6140, 1996.

[Mir96] Kalim U. Mir. A restricted genetic alphabet for DNA computing. In *Proceedings of the Second Annual Meeting on DNA Based Computers, held at Princeton University, June 10-12, 1996.*, DIMACS: Series in Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, June 1996.

[MLRS00] C. Mao, T. H. LaBean, J. H. Reif, and N. C. Seeman. Logical computation using algorithmic self-assembly of dna triple-crossover molecules. *Nature*, 407:493–6, 2000.

[NMW99] C. G Nevill-Manning and I. H. Witten. Protein is incompressible. In *IEEE Data Compression Conference*, pages 257–266. IEEE Computer Society TCC, 1999.

[OGB97] Martin Orlian, Frank Guarnieri, and Carter Bancroft. Parallel primer extension horizontal chain reactions as a paradigm of parallel DNA-based computation. In David Wood, editor, *Proceedings of the 3rd DIMACS Workshop on DNA Based Computers, held at the University of Pennsylvania, June 23 – 25, 1997*, DIMACS: Series in Discrete Mathematics and Theoretical Computer Science., pages 142–158, Providence, RI, June 1997. American Mathematical Society.

[PSS$^+$94] A. C. Pease, D. Solas, E. J. Sullivan, M. T. Cronin, C. P. Holmes, and S. P. Fodor. Light-generated oligonucleotide arrays for rapid DNA sequence analysis. *Proc. Natl Acad. Sci. USA*, 91:5022–5026, 1994.

[RDG$^+$97] J. A. Rose, R. Deaton, M. Garzon, R. C. Murphy, D. R. Franceschetti, and S. E. Stevens, Jr. The effect of uniform melting temperatures on the efficiency of DNA computing. In David Wood, editor, *Proceedings of the 3rd DIMACS Workshop on DNA Based Computers, held at the University of Pennsylvania, June 23 – 25, 1997*, DIMACS: Series in Discrete Mathematics and Theoretical Computer Science., pages 35–42, Providence, RI, June 1997. American Mathematical Society.

[Rei97] John H. Reif. Local parallel biomolecular computing. In David Wood, editor, *Proceedings of the 3rd DIMACS Workshop on DNA Based Computers, held at the University of Pennsylvania, June 23 – 25, 1997*, DIMACS: Series in Discrete Mathematics and Theoretical Computer Science., pages 243–264, Providence, RI, June 1997. American Mathematical Society.

[Rei98a] John Reif. Parallel molecular computation: Models and simulations. *Algorithmica, Special Issue on Computational Biology*, 1998.

[Rei98b] John H. Reif. Paradigms for biomolecular computation. In C. S. Calude, J. Casti, and M. J. Dinneen, editors, *Unconventional Models of Computation*. Springer, 1998.

[Rob94] S. S. Roberts. Turbocharged pcr. *Jour. of N.I.H. Research*, 6:46–82, 1994.

[RWB$^+$96] Sam Roweis, Erik Winfree, Richard Burgoyne, Nickolas V. Chelyapov, Myron F. Goodman, Paul W. K. Rothemund, and Leonard M. Adleman. A sticker based architecture for DNA computation. In *Proceedings of the Second Annual Meeting on DNA Based Computers, held at Princeton University, June 10-12, 1996.*, volume 44 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science.*, pages 1 – 30. American Mathematical Society, May 1996.

[Sch96] Bruce Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* John Wiley & Sons, Inc, 1996.

[Sto88] James A. Storer. *Data Compression: Methods and Theory.* Computer Science Press, 1988.

[Suy98] A. Suyama. Dna chips - integrated chemical circuits for dna diagnosis and dna computers, 1998.

[TRB99] C. T. Taylor, V. Risca, and C. Bancroft. Hiding messages in DNA microdots. *Nature*, 399:533–534, 1999.

[Win95a] Erik Winfree. Complexity of restricted and unrestricted models of molecular computation. In Eric B. Baum and Richard J. Lipton, editors, *DNA Based Computers*, volume 27 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science.*, pages 187–198. American Mathematical Society, May 1995.

[Win95b] Erik Winfree. On the computational power of DNA annealing and ligation. In Eric B. Baum and Richard J. Lipton, editors, *DNA Based Computers*, volume 27 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science.* American Mathematical Society, May 1995.

[Win98] Erik Winfree. Simulations of computing by self-assembly. In *Proceedings of the Fourth DIMACS Meeting on DNA Based Computing*, pages 213–242, May 1998.

[WLWS98] E. Winfree, F. Liu, Lisa A. Wenzler, and N. C. Seeman. Design and self- assembly of two dimensional dna crystals. *Nature*, 394:539–544, 1998.

[WYS96] Erik Winfree, Xiaoping Yang, and Nadrian C. Seeman. Universal computation via self-assembly of DNA: Some theory and experiments. In *Proceedings of the Second Annual Meeting on DNA Based Computers, held at Princeton University, June 10-12, 1996.*, volume 44 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science.*, pages 191 – 214. American Mathematical Society, May 1996.

[ZL77] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Trans. Inf. Theory*, IT-23:337–343, May 1977.