

An Overview of the Evolutionary Trends in Molecular Computing using DNA

Abhinav Maurya

Veermata Jijabai Technological Institute, India.

abhinav.maurya@yahoo.com

Anu Nair

Veermata Jijabai Technological Institute, India.

anuv.nair@gmail.com

Sugata Sanyal

Tata Institute of Fundamental Research, India.

sanyal@tifr.res.in

Abstract

The paper seeks to address the incipient paradigm of using the biochemical molecule of DNA to solve computational problems.

DNA computers use strands of DNA (deoxyribonucleic acid) to perform computing operations. The computer consists of two types of strands - the instruction strands and the input data strands. The instruction strands splice together the input data strands to generate the desired output data strand.

Since the instruction strands are generic nucleotide codes for the various operations, DNA computers meet the accepted standard for being classed as true computers according to the Turing machine concept.

A strong point in favor of DNA computing outlined by the experts is the potential storage capacity of DNA, which far exceeds that of the most advanced silicon storage devices.

A major challenge that DNA circuit engineers face is the difficulty of predicting circuit performance at the design stage, with the consequence that actual construction requires significant experimental effort, even for very simple circuits. To address this fundamental obstacle, evolutionary simulations have been used to create new circuit components and optimize circuit performance of DNA computing devices.

Keywords

DNA computers, DNA chips, Genetic Algorithms, Cellular logic, DNA data storage.

1. Introduction

DNA computers, an innovation with the potential to address the short-comings of silicon-based computers, are radically different from what most people recognize as a computer. Research on DNA computers is timely since silicon devices have reached their zenith for miniaturization. In 1959, R. Feynman described the possibility of building computers which were sub-microscopic [1]. The statement is not yet true. Though the claims made by G. Moore pertaining to the increase in the chip transistor density by a factor of two every twelve months (later rectified to eighteen months) have held true to this day, the possibility of further shrinking the size of transistors implies the fabrication of transistors out of groups of atoms, which present inevitable hardships. Thus, it is clear that the solution to increasing the processing power of computers is not to reduce the distance between sequential processing elements by using superscalar integration but by harnessing the capacity for concurrent processing.

DNA computers, unlike silicon-based computers are created on the assumption of generating a mammoth number of problem instances and using the hydrogen bonding possible between the two affine pairs of nucleotides to solve them.

DNA computing has the potential to overcome the limits imposed on the processing power of silicon devices by Heisenberg's Uncertainty Principle [2]. This paper sets the stage for considering this topic by discussing this alternative paradigm of using the biochemical macro-molecule of DNA to solve computational problems. It then focuses on DNA computing in all its varieties and considers the benefits

and possible problems of this radically different form of computing.

DNA directs the synthesis of proteins, which are molecules made up of amino acids arranged in specific linear sequences. As there are 20 different amino acids, it is necessary for DNA to provide a code for each of these. If 3 successive bases are used to specify 1 amino acid, then the total number of combinations would be 64. These 64 possible triplet combinations of bases which contribute a code for amino acids are known as codons. Since there are only 20 amino acids and 64 possible codons, most amino acids are specified by more than 1 code, hence the code is said to be degenerate.

2. Limitations of Conventional Computing Technology

First, let us look at the limitations imposed by physics. The traditional computing design paradigm focuses on decreasing the distance that information (in the form of electrical signals) has to travel; in other words, shortening the distance between processing elements (PE's). This has meant packing more and more processing elements or transistors into the central processing chip of the computer. Today, this packing of transistors has reached an amazing level of density. As relentless as this progress has been thus far, it is dependent upon continuing to find new means of integration. In the coming years, transistors will have decreased in size to such a great extent that the only way to make them smaller is to construct them out of individual atoms or small groupings of atoms. Unfortunately, quantum effects of physics operating on that size and scale will prevent the effective transmission of signals. The Heisenberg Uncertainty Principle states that particles (such as the electrons that make up the information signals that flow through computers) can exhibit the strange behavior of position uncertainty due to their subatomic-scale mass.

Given that the circuit pathways of computers must be able to reliably transmit information; this quantum effect that emerges at the atomic scale is clearly a problem. As miniaturization continues, it will force silicon-based computers up to their design limits.

There are two other problems with silicon-based computers. First, the components out of which computer processing chips are made are toxic (e.g. arsenic which is used for doping) and therefore present challenges in both fabrication and disposal. Second, silicon-based computers are not very energy efficient. They waste a great deal of energy in the form of the heat that they generate and the energy they consume.

3. DNA Computing

DNA (deoxyribonucleic acid) molecule has a double-helical structure composed of two sugar-phosphate backbones formed by the polymerization of deoxyribose sugar. Placed between the two backbones are pairs of the nucleotides Adenine, Cytosine, Guanine, and Thymine. Due to the spatial precondition of the nucleotide pairs having to fit between the two polymer backbones, the only nucleotide pairs that can possibly form hydrogen bonds in the DNA molecule are Adenine-Thymine and Cytosine-Guanine.

DNA computers use single strands of DNA to perform computing operations. The computer consists of two types of strands - the instruction strands and the input data strands. Due to the tendency of singular DNA strands to form a double-helical structure in the case of their nucleotide sequences being complementary to each other, instruction strands splice together the input data strands to generate the desired output data strand. Since the instruction strands are generic nucleotide codes for the various operations, DNA computers meet the accepted standard for being classed as true computers according to the Turing machine concept [3].

DNA computing focuses on the use of massive parallelism, or the allocation of tiny portions of a computing task to many different processing elements. The structure of the DNA allows the elements of the problem to be represented in a form that is analogous to the binary code structure (1's and 0's) which characterizes the most basic form of computer languages. Trillions of unique strands of DNA (encoded in a quaternary number system using the four nucleotides A, C, T, and G) are able to represent all of the possible solutions to a problem. The 'possible solution' strands are allowed to react with the 'problem' strands. Given the binding rules for nucleotides present in the DNA, the strands that complement one another will bind, yielding a collection of whole DNA molecules that contain the solutions. A number of processing steps are performed on the resulting mixture to isolate a correct solution from all the possibilities. The results are then analyzed by the electronic portion of the computer.

The ability of trillions of reactions to occur simultaneously provides the equivalent of massively parallel processing in silicon-based computers, in which a huge number of possible problem solutions or searches through pools of information for the answer are performed at the same time.

4. The Genesis of DNA Computing

The first major breakthrough in the field of DNA

computing occurred in 1994 when Adleman used DNA computing to solve the Traveling Salesman Problem which is formally known as the directed Hamiltonian problem.

The strategy employed by Adleman [4] is briefly explained here. Each city is encoded as a unique sequence of nucleotides. In addition, the nucleotide sequence for each route is constructed as follows: the first half of the route nucleotide sequence is complementary to the second half of the nucleotide sequence of the first city, whereas the second half of the route nucleotide sequence is complementary to the first half of the nucleotide sequence of the second city. As a result, when these strands interact in a solution, the two city strands in question link up with their corresponding complementary halves of the route connecting them. The remaining halves of the two linked city strands further link up with other city strands via other route strands present in the solution, leading to the formation of a multi-link DNA strand which represents a path in the graph of the Traveling Salesman Problem. Since there exist trillions of copies of each city and route strand, the computation of all possible paths in the graph proceeds in parallel, leading to constant-time computation of paths with an upper bound on their length in terms of the individual city and route strands. As a result, optimal paths can be calculated within feasible time even in a graph having a large number of nodes.

Once the computation of all possible paths in the graph has been performed, we employ a technique known as the *Polymerase Chain Reaction (PCR)* [5] using the DNA polymerase enzyme which allows us to iteratively replicate a specific sequence of DNA, increasing the concentration of the DNA strand with the desired starting and ending cities. After many iterations of PCR, the DNA being worked on is amplified exponentially. So to selectively amplify the itineraries that start and stop with the cities of interest, primers that are complimentary to the cities are used. After PCR, we obtain a test tube full of double-stranded DNA of various lengths, encoding itineraries that start with the salesman's starting city and end with the city he wants to visit last.

Thereafter, using *Gel Electrophoresis* which can resolve amongst DNA strands of different sizes, the paths with the number of cities on the salesman's itinerary are segregated, and a technique known as *affinity purification* is used which in each step selects only those paths which pass through a city mentioned on the itinerary. Finally, using the technique of *graduated PCR*, we do a series of PCR amplifications using the primer corresponding to the first city and other cities on the itinerary. For each of the primer pairs, the nucleotide length of the PCR product is found out which helps us to determine the position of the corresponding city on the path taken by the salesman.

5. Forays into DNA Computing

In March, 2002, NASA announced that a team led by Adleman developed a DNA computer that solved a problem that required evaluating one million alternatives for their ability to meet 24 separate criteria. Work, such as that done by Adleman, is aimed at increasing the complexity of problems that such computers can solve, while also developing measures to reduce the levels of errors in the process. Not all DNA computing adheres to the format established by Adleman's work. A variant of DNA computing is being designed to emulate more closely the structure of conventional computers. Scientists at the University of Rochester in 1997 developed logic gates made of DNA. Instead of using electrical signals to perform logical operations, DNA logic gates rely on inputs and outputs of DNA. In November of 2001, scientists from Israel's Weizmann Institute developed a series of tiny DNA computers, trillions of which can fit in solution in a test tube. These computers use a form of software, also made of DNA, and can compute with high reliability.

6. DNA Chips

A closely related technology is the DNA chip. Such a device is similar in principle to DNA computers but is simpler in structure. The basic concept involves the use of large numbers of DNA strands, which are bound to tiny glass chips. Each of these strands, or probes, is able to bind to sample sequences of genetic material. These chips allow researchers to evaluate thousands of genetic sequences simultaneously, for use in developing treatments for disease. Research into the continued improvement and additional applications of DNA chips is occurring at many universities [6].

7. Genetic Algorithms

DNA computing employs a crude form of parallelized brute-force approach to solving problems. A more refined approach in this connection would be genetic programming which explicitly use the power of parallelization to pursue all probable solutions concurrently, and hones in on the most optimal one by comparison of certain parameters amongst the final candidate solutions.

Genetic algorithm [7] is any type of software that uses variation and selection to produce an output optimized by generations of virtual evolution [8]. That output may be a program, a value, or even a picture. The genetic algorithm needs a process for generating new

variants and feedback, or fitness criteria, in order to determine which variants to discard and which to use.

The genetic algorithm is often mentioned in connection with Artificial Life (Alife) [9], which is the study of virtual organisms created in a computer. These virtual organisms usually live on a virtual grid, and sometimes even reproduce with each other and consume virtual nutrients. These Alife simulations are occasionally used to study real life forms, or to evolve behaviors shared with real life forms. By creating artificial evolutionary environments using genetic algorithms, we can try to trace the causes of various phenomena like evolution, diseases, intelligent autonomous agents, etc. [9]

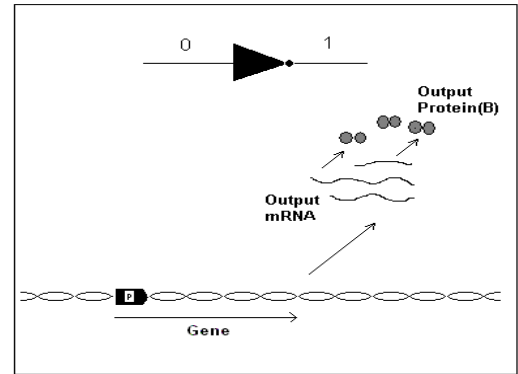
In one Alife simulation, "organisms" are merely lit-up pixels, which influence pixels around them based on whether adjacent pixels are lit up or not. Surprisingly, the interactions between these organisms can create complex high-level effects that no one could have predicted by looking at the low-level components. Such effects can be used to efficiently implement algorithms in Image Processing.

Because genetic algorithms may explore large portions of the search space to a particular problem, they may yield solutions that human operators never would have thought up. The downside to these programs is that for problems with a very large search space, the number of possible solutions may be huge. Therefore, the system can be computationally intensive. By uniting the power of human thought with the brute force search-and-test randomness of genetic algorithms, we can create unique new solutions to problems in biology, engineering, mathematics, and other fields.

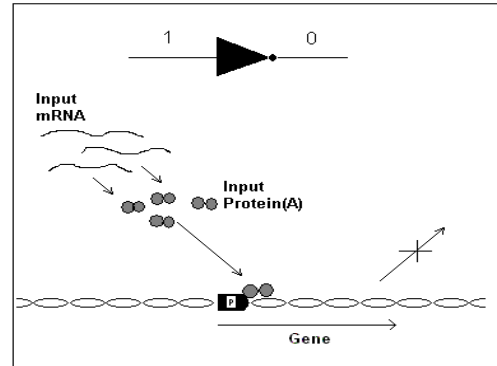
8. Digital Cells

In the past few years, scientists have taken the first steps towards creating a host of cellular robots that are programmed to carry out tasks such as detecting and cleaning up environmental pollutants, tracking down cancer cells in a body, and manufacturing antibiotics or molecular-scale electronic components. The notions of electrical engineering—digital logic, memory, and oscillators—have been borrowed into the realm of biology. The idea is to create cells with computer programs hardwired into the DNA. Eventually, the goal is to produce genetic applets, little programs that can be downloaded into a cell simply by incorporating DNA into it. By cutting and pasting pieces of genetic material, and most recently using artificial evolution as a design tool, engineers have started to program microbes to carry out programmed behaviors.

Silicon circuits perform complex operations using a handful of simple components known as logic gates. Genetic-circuit engineers are now building similar devices inside cells.



(a)



(b)

Fig.1 A Cellular Inverter

Between 1998 and 2001, Weiss [12] took some of the first steps toward building cellular logic gates when he modeled and built a cellular inverter, an AND gate and two other gates. In Weiss' inverter [Fig. 1], the input bit and output bit are encoded in the concentrations of two proteins—for simplicity's sake, call them protein A and protein B. If the concentration of protein A is high, the input bit is a 1 (Fig 1(a)); if the concentration is low, the input bit is a 0 (Fig 1(b)). Similarly, the concentration of B corresponds to the output bit. Weiss constructed in *Escherichia coli* bacteria a loop of DNA containing two important pieces: the gene with the instructions for building protein B, and near that, a segment of DNA to which protein A binds.

To make protein B, a special molecule called messenger RNA assembles itself along the DNA, copying the gene's instructions and carrying them to the cell's protein-making factory. If the concentration of A is

high, molecules of protein A will bind to the DNA loop and block the messenger RNA from attaching to the DNA. This prevents the cell from building protein B. If, on the other hand, the concentration of A is low, then protein B will be built in abundance. Thus, in Weiss' circuit, B is high when A is low, and vice versa. Weiss' other gates are constructed in similar, slightly more complicated ways.

By hooking together inverters, engineers can create a wide variety of interesting devices. In 2000, Gardner and his colleagues Collins and Cantor [13], all from Boston University, built a memory device in *E. coli* out of two inverters for which the output protein of one is the input protein of the other, and vice versa. In the same year, Elowitz and Leibler of Rockefeller University in New York City made an oscillator [14] in which three inverters are connected in a loop so that the output protein of each inverter is the input protein of the next inverter. In one test of their system, a fluorescent protein became active whenever one of the proteins was in its low state.

9. DNA Data Storage

The other strong point in favor of biological computing outlined by the experts is the potential storage capacity of DNA [11] within bacteria. Experiments have successfully retrieved 57-99 base pairs of non-native information from a bacterium. Considering that a milliliter of liquid can contain 10^9 bacteria, if we assume an average of 80 base pairs of information per bacterium, using 4 base pairs per byte that would give 19 Gigabytes of data for a volume of one milliliter. In comparison, the current areal density record for magnetic storage stands at 10 Gigabytes per square inch. If we consider that the hard-disk platter is 1cm thick that gives 1.5 Gigabytes per milliliter. Therefore the storage density of DNA within bacteria fares strongly against traditional storage mediums [16].

As the amount of stored data grows, so does the overhead needed to keep the data in order. For 10^9 bacteria, 30 bits or 15 base pairs are needed to uniquely identify each bacterium. Even though the rate of growth for the needed sequence number space is $O(\log_2 n)$, this turns out to be a significant percentage of the current usable capacity.

Another practical challenge that needs to be tackled is the issue of speed. The observations made during the experiments is it took two hours to retrieve the stored information, but do not indicate how long it took to store the information.

10. Possibilities of Glitches

It is far easier to describe the schematics of these circuits than to build them for operation inside a cell. For instance, to hook up one gate to the next, the amount of protein produced by the first gate must be the right amount to activate the next gate. And at every step, the output protein must be either very high or very low, to avoid false positives or negatives.

Making such adjustments via trial and error is prohibitively expensive. Genetic-circuit engineers usually use enzymes to cut pieces of DNA out of one organism's genome, glue the pieces together in different combinations using other enzymes, and then put them in another organism. Because the circuits generally include complicated feedback loops, the equations that model the reactions tend to be nonlinear.

Because many of the parameters of the biochemical reactions are only partly understood, and because the random jiggling of molecules complicates the picture, mathematical simulations give only partial information about whether a circuit will work. Consequently, it is quite common to build a circuit only to find, for instance, that it produces 50 protein molecules when you really need 500.

11. Benefits of DNA Computing

The various forms of DNA computing possess a number of benefits over their silicon-based counterparts. There are five main benefits that DNA computing can offer.

First, a biological computing process can be up to one billion times more energy efficient than the computational processes of silicon-based computers.

Second, DNA is capable of storing an astounding amount of information for a given volume.

Third, the component materials are plentiful in supply and easily obtained in most cases. Also, the components, e.g. DNA, are non-toxic.

Fourth, biological computing allows for massively parallel computing. Experts believe that it should be possible to produce massively parallel processing in biological computers at a level of 10^{17} operations per second or more, or a level that silicon-based computers will never be able to match.

Fifth, computing devices that are living or composed of living components have the potential to share two characteristics that allow living organisms to adapt so well to changing conditions: the capability of healing injuries and the capacity to self-improve.

12. Potential Problems with Biological Computing.

A number of problems with biological computing must be resolved before it can reach its full potential.

First, in some cases the types of genetic sequences that would have to be synthesized to make full functioned genetic “applets” or genetic “robots” possible would be prohibitively expensive using current methods.

Second, despite their capacity for massively parallel calculations, the individual operations of DNA computers are quite slow in comparison to those of their silicon-based counterparts.

Third, DNA computing requires quantities of DNA that can only be used once, as reuse can contaminate reaction vessels and lead to less accurate results.

Fourth, DNA computing is prone to errors at a level that would be considered unacceptable by the silicon-based computing industry.

Finally, biological computing raises problematic ethical and moral concerns. The idea of scientists being able to create a living creature raises a number of troubling questions.

13. Conclusion

In a number of forms, DNA computing is already being used outside of research labs. DNA chips for analysis of genetic sequences in pharmaceutical development have been commercially available since the late 1990's. Further, Olympus Optical unveiled a commercially practical DNA computer for gene analysis in February, 2002.

As with electronics, the real power will come from assembling cellular logic gates into large circuits. To do that, many technical challenges must be overcome. For example, each new gate in a circuit must usually be turned on and off by different proteins than those that control the previous gates. Scientists are now pushing boundaries to utilize the potential contained in the most fundamental units of life and the biological world to answer the limitations imposed by the laws of physical computing.

14. References

- [1] Feynman R. P., *Miniaturization*, Reinhold Publishing Corporation, New York, pp 282-296, 1961.
- [2] W. Heisenberg, "Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik", *Zeitschrift für Physik*, 43 1927, pp. 172-198. English translation: J. A. Wheeler and H. Zurek, *Quantum Theory and Measurement* Princeton Univ. Press, 1983, pp. 62-84
- [3] Turing A. M., *On Computable Numbers, with an Application to the Entscheidungs problem*, 1936, <http://www.abelard.org/turpap2/turpap2.htm>
- [4] Adleman, L.M., *Molecular Computation of Solutions to Combinatorial Problems*, *Science*, 266: 1021-1024, November 11, 1994.
- [5] Powledge T., *The Polymerase Chain Reaction*, <http://www.faseb.org/opa/bloodsupply/pcr.html>
- [6] Schena M, Shalon D, Davis RW, Brown PO. (1995), Quantitative monitoring of gene expression patterns with a complementary DNA microarray *Science*. Oct 20; 270 (5235): 467-70
- [7] Belew R. K. and Vose M. D., editors, *Foundations of Genetic Algorithms 4*, pages 117 – 139, Morgan Kaufmann, 1997.
- [8] Corne D., and Shapiro J. L., editors. *Evolutionary Computing*, volume 1305 of *Lecture Notes in Computer Science*, Springer, 1997.
- [9] Levy S., *Artificial Life*, ISBN: 0224035991, Cape, 1992.
- [10] Weiss R. & Basu S., *The Device Physics of Cellular Logic Gates, First Workshop on Non-Silicon Computing, Cambridge, Massachusetts*, 54-61, Feb 2002, <http://www.hpcaconf.org/hpca8/>
- [11] Gardner T., Cantor R., and Collins J., *Construction of a genetic toggle switch in Escherichia coli*, *Nature*, 403:339-342, January 2000.
- [12] Elowitz M. and Leibler S., *A synthetic oscillatory network of transcriptional regulators*, *Nature*, 403:335-338, January 2000.
- [13] Wong, Wong, Foote, *Organic Data Memory - Using the DNA Approach*, Comm. ACM, 2003.
- [14] New Scientist, *Data Stored in multiplying bacteria*, 2003, <http://www.newscientist.com/news/news.jsp?id=ns99993243>