IMPLEMENTATION OF BOOLEAN CIRCUITS USING DNA COMPUTING

Ashish Lamaniya[1], Brajesh Patel[2]
[1]Department of Computer Sc. & Engineering, SRIT, RGPV University, Jabalpur, India
[2]Head of Computer Sc. & Engineering Department, SRIT, RGPV University, Jabalpur, India.

ABSTRACT

DNA-based computing is at the intersection of several threads of research. The information bearing capability of DNA molecules is a cornerstone of modern theories of genetics and molecular biology. The information in a DNA molecule is contained in the sequence of nucleotide bases, which hydrogen bond in a complementary fashion to form double-stranded molecules from single-stranded oligonucleotides. Various aspects of life inspired early results in computer science in the 1950's (J. von Neumann's universal constructor and computer, S. Ulam's models of growth using cellular automata). A second development occurred in the early 1970's with J. Holland's computational implementation of fundamental biological mechanisms, such as genetic operations (splicing, recombination and mutation) and evolution. Finally, a third stage inaugurated by L. Adleman's 1994 proof of concept that recombinant properties of real DNA can actually use massive parallelism to solve problems appropriately encoded into single DNA strands. DNA computing is an old but a developing technology. Vast range of applications can be developed using DNA, including very complex problems like NP-complete problems. The versatile utilities inspire me to select this field as my research work. In my work I have implemented of different basic logic gate and some combinational circuits.

KEYWORDS: *Digital Circuits, Combinational Circuits, DNA, Boolean algebra, NP Problems.*

1. INTRODUCTION

The manipulation of complex DNA solutions with genetic engineering tools has been proposed recently as a chemical methodology for solving instances of computationally intractable (so- called NP-complete) combinatorial problems [3]. DNA computing holds out the promise of important and significant connections between computers and living systems, as well as promising massively parallel computations. Before these promises are filled, however, important challenges related to errors and practicality has to be addressed. On the other hand, new directions toward a synthesis of molecular evolution and DNA computing might circumvent the problems that have hindered development, so far.

Boolean circuits are defined in terms of the logic gates they contain. For example, a circuit might contain binary AND and OR gates and unary NOT gates, or be entirely described by binary NAND gates. Each gate corresponds to some Boolean function that takes a fixed number of bits as input and outputs a single bit.

DNA computing is an old but a developing technology. Vast range of applications can be developed using DNA, including very complex problems like NP-complete problems. The versatile utilities inspire me to select this field as my research work. During literature survey I found rear work is done in the field of Boolean circuits through DNA computing. Some theoretical concepts had been discussed by some researches, and that were only in elementary level. My objective is to give the implementation of Boolean circuits using this emerging technique of computing. I focused on various Logic gates and then the combinational circuits.

2. DNA COMPUTING FUNDAMENTALS

Deoxyribonucleic acid is the expansion of the abbreviation DNA which is the germ plasma of all the living types. It is a macromolecule of biology which is made up of many small nucleotides. And in that nucleotide, it is composed of a unique base out of the four varied types of it. The four bases are adenine (A), thymine (T) or guanine (G) and cytosine (C) matching to the corresponding nucleotide. The single-stranded DNA is developed with positioning of one end known as (5 prime) 5′ and the location of the other end is said to be (3 prime) 3′ [2].

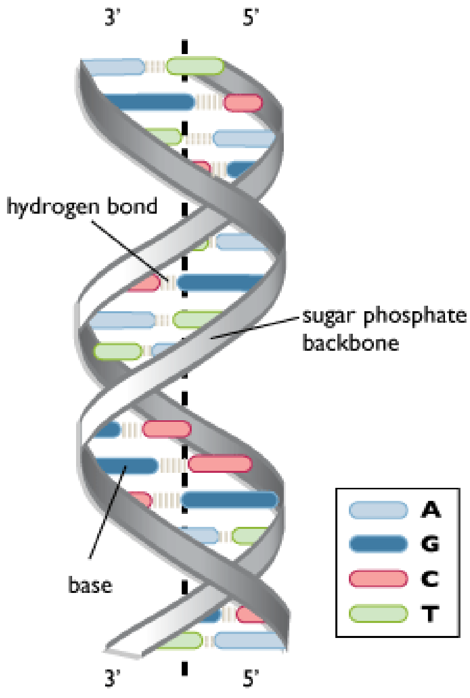


Figure.1: The DNA

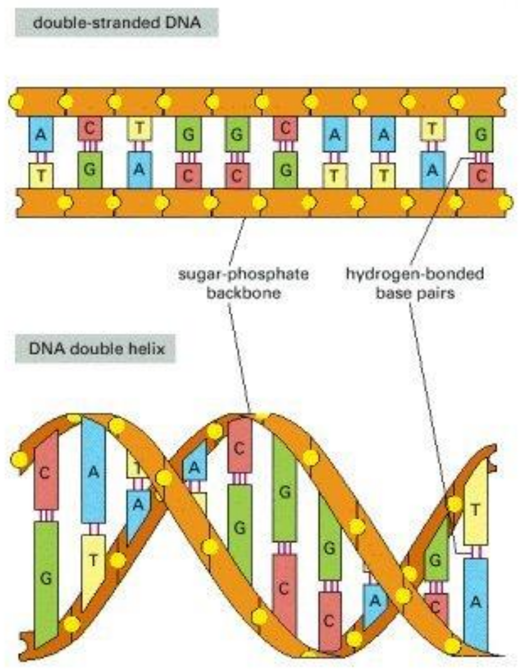


Figure 2: DNA Double Helix Structure

Naturally, the DNA is in the form of double helical structure or it can be said that it is a double strand molecule. The two individual complementary strands of DNA are joined together, by making a bond with the complementary (A and T or C and G) bases with the help of hydrogen bond between them for bonding. This is done to form the double-helix structure of DNA.

The biological instruction present in the strand of DNA is stored as a code which is determined by the order, or sequence, of these nitrogen bases, further used to determines the information necessary for building and maintaining an organism, similar to the way of forming words and sentences using letters of the alphabet in a certain order. For example, the sequence ATCGTT might instruct for blue eyes, while ATCGCT might instruct for brown.

2.1 DNA Computers

DNA computing performs the computations using biological molecules, rather than traditional silicon chips. DNA (deoxyribonucleic acid) molecules, the material our genes are made of, have the potential to perform calculations many times faster than the world's most powerful human-built computers. DNA might one day be integrated into a computer chip to create a so-called biochip that will push computers even faster. DNA molecules have already been harnessed to perform complex mathematical problems. While still in their infancy, DNA computers will be capable of storing billions of times more data than your personal computer.

DNA computers can't be found at your local electronics store yet. The technology is still in development, and didn't even exist as a concept a decade ago. In 1994, Leonard Adleman introduced the idea of using DNA to solve complex mathematical problems. Adleman, a computer scientist at the University of Southern California, came to the conclusion that DNA had computational potential after reading the book "Molecular Biology of the Gene," written by James Watson, who co-discovered the

structure of DNA in 1953. In fact, DNA is very similar to a computer hard drive in how it stores permanent information about your genes.
Adleman is often called the inventor of DNA computers. His article in a 1994 issue of the journal Science outlined how to use DNA to solve a well-known mathematical problem, called the directed Hamilton Path problem, also known as the "traveling salesman" problem.
The success of the Adleman DNA computer proves that DNA can be used to calculate complex mathematical problems. However, this early DNA computer is far from challenging silicon-based computers in terms of speed. The Adleman DNA computer created a group of possible answers very quickly, but it took days for Adleman to narrow down the possibilities. Another drawback of his DNA computer is that it requires human assistance. The goal of the DNA computing field is to create a device that can work independent of human involvement.
DNA computer components -- logic gates and biochips -- will take years to develop into a practical, workable DNA computer. If such a computer is ever built, scientists say that it will be more compact, accurate and efficient than conventional computers.

2.2 The Adleman Experiment

As I earlier mentioned that the Adleman use DNA to solve the directed Hamilton Path problem (traveling salesman problem). In following sections I gave the summary of the Adleman experiment.
Suppose that I live in LA, and need to visit four cities: Houston, Chicago, Miami, and NY, with NY being my final destination. The airline I'm taking has a specific set of connecting flights that restrict which routes I can take (i.e. there is a flight from L.A. to Chicago, but no flight from Miami to Chicago). What should my itinerary be if I want to visit each city only once?

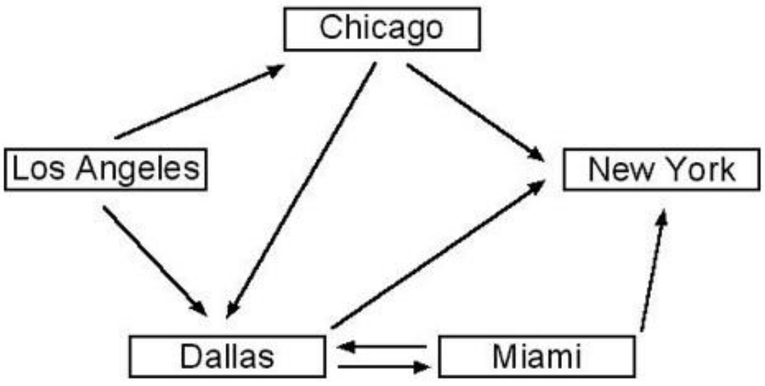


Figure 3: Five City Problem

The method Adleman used to solve this problem is basically the shotgun approach. He first generated all the possible itineraries and then selected the correct itinerary. This is the advantage of DNA. It's small and there are combinatorial techniques that can quickly generate many different data strings. Since the enzymes work on many DNA molecules at once, the selection process is massively parallel[6].

2.3 The DNA Computation Model

To put DNA computation on a concrete framework, the article by Lila Kari breaks the process into smaller steps that can be regarded as primitive operations for the DNA computer [3].

1. **Synthesizing:** This stage involves creating a single DNA strand consisting of polynomially many base pairs.
2. **Mixing:** This step involves taking the contents of two test tubes and mixing them together in a third. This step may seem frivolous, but becomes important in the theoretical framework.
3. **Annealing:** This process, also known as hybridization, involves lowering the temperature of the solution so that two DNA sequences to attach together in a double helix.
4. **Amplifying:** A reaction known as the Polymerase Chain Reaction (PCR) allows one to produce a copy of a DNA strand. Kari observes that exponentially many strands can be produced by repeatedly performing this operation in parallel.
5. **Separating:** In this step, gel electrophoresis is used to determine the length of a DNA strand.

6. **Extracting:** A step called affinity purification allows one to find strands that match a given substring of DNA.
7. **Cutting:** Certain enzymes allow one to cut DNA at sites with particular patterns of bases.
8. **Ligating:** This is the opposite of cutting; certain enzymes provide for the ability to connect DNA strands with certain endings.
9. **Substituting:** This fairly complex operation allows for insertions, deletions, or substitutions of sequences of base pairs in a DNA strand [B].
10. **Detecting and Reading:** Once a DNA sequence is present in solution, this stage involves determining the sequence of base pairs that compose that strand of DNA in order.

3. BOOLEAN ALGEBRA & CIRCUITS

In mathematics and mathematical logic, Boolean algebra is the subarea of algebra in which the values of the variables are the truth values *true* and *false*, usually denoted 1 and 0 respectively. Instead of elementary algebra where the values of the variables are numbers and the main operations are addition and multiplication, the main operations of Boolean algebra are the conjunction *and*, denoted ∧, the disjunction *or*, denoted ∨, and the negation *not*, denoted ¬. Boolean algebra has been fundamental in the development of computer science and digital logic. It is also used in set theory and statistics. The basic operations of Boolean algebra are the following ones:

- **And** (conjunction), denoted $x \wedge y$ (sometimes x AND y or Kxy), satisfies $x \wedge y = 1$ if $x = y = 1$ and $x \wedge y = 0$ otherwise.
- **Or** (disjunction), denoted $x \vee y$ (sometimes x OR y or Axy), satisfies $x \vee y = 0$ if $x = y = 0$ and $x \vee y = 1$ otherwise.
- **Not** (negation), denoted $\neg x$ (sometimes NOT x, Nx or !x), satisfies $\neg x = 0$ if $x = 1$ and $\neg x = 1$ if $x = 0$.

A logic gate is an elementary building block of a digital circuit. Most logic gates have two inputs and one output. At any given moment, every terminal is in one of the two binary conditions *low* (0) or *high* (1), represented by different voltage levels. The logic state of a terminal can, and generally does, change often, as the circuit processes data. In most logic gates, the low state is approximately zero volts (0 V), while the high state is approximately five volts positive (+5 V). Some circuits may have only a few logic gates, while others, such as microprocessors, may have millions of them. There are seven basic logic gates: AND, OR, XOR, NOT, NAND, NOR, and XNOR.

1. **AND -** True if A and B are both True
2. **OR -** True if either A or B are True
3. **NOT -** Inverts value: True if input is False; False if input is True
4. **XOR -** True if either A or B are True, but False if both are True
5. **NAND -** AND followed by NOT: False only if A and B are both True
6. **NOR -** OR followed by NOT: True only if A and B are both False
7. **XNOR -** XOR followed by NOT: True if A and B are both True or both False

4. DNA COMPUTING SOFTWARE

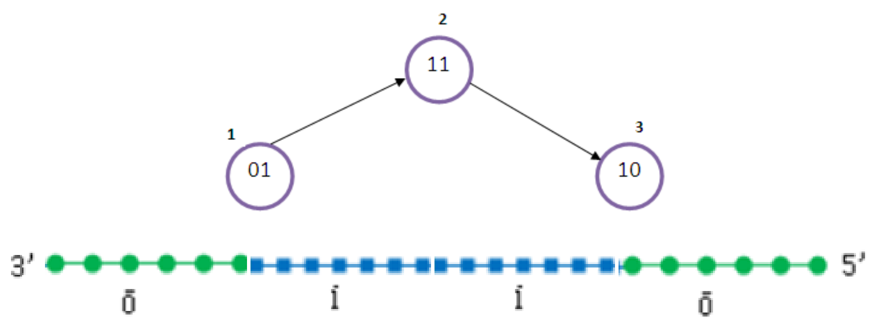
The use of true DNA computers may not be possible, as there don't exist commercially at the moment. It seems perfectly reasonable to try to build software to simulate one. That was the motivating idea for this program. The purpose of this program is only to help designing algorithms for DNA computers. The program uses an ideal model for the DNA molecules. The model doesn't take into account the possibilities of imperfect operations, error correction etc. Those are research questions and need to be taken care of separately. Also the biological, chemical and physical factors affecting the calculations are not covered by this program as they need much more specialized knowledge and expertise in the areas of molecular biology and biochemistry.

This program is a C++ that provides the tools to perform DNA calculations using conventional electronic computers. Library functions include interface functions to perform the computations and internal functions to implement the operations.

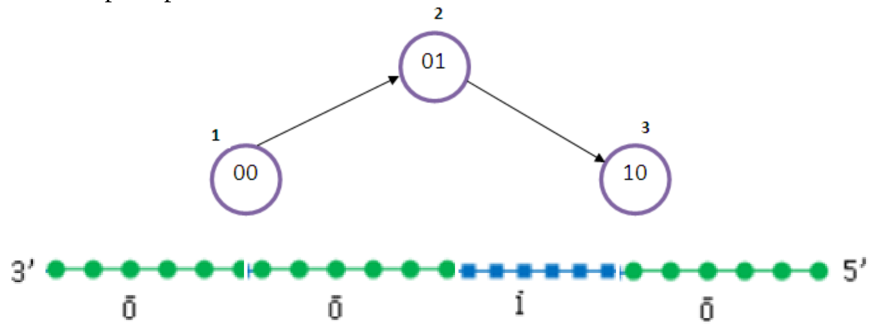
4.1 System Implementation

The system implementation of basic building blocks of Boolean circuit is shown in following section by displaying the graph representation of Boolean elements and their corresponding DNA strands.

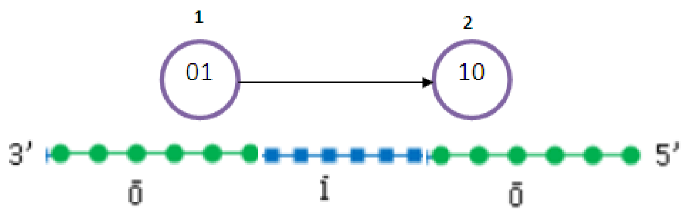
OR Gate: The Graph representation and DNA strand for the OR are as follows:



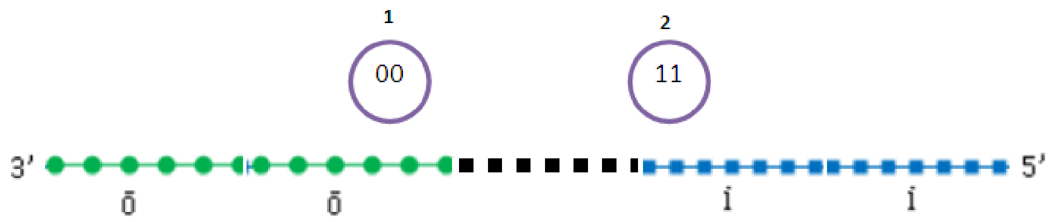
NAND Gate: The Graph representation and DNA strand for the NAND are as follows:



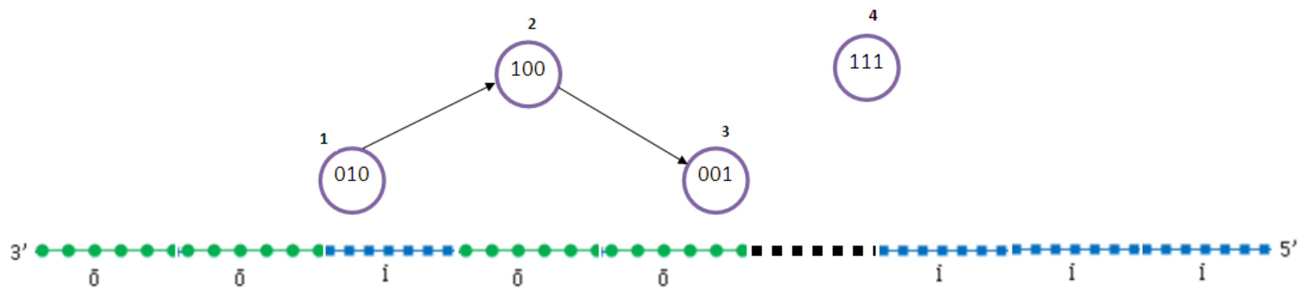
XOR Gate: The Graph representation and DNA strand for the XOR are as follows:



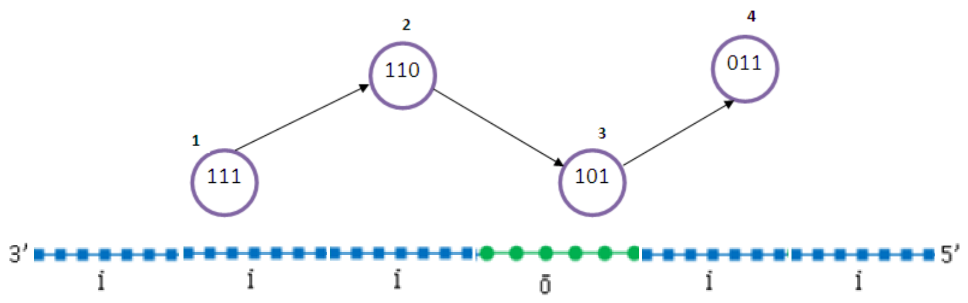
XNOR Gate: The Graph representation and DNA strand for the XNOR are as follows:



Full Adder: Graph representation and DNA strand for the Sum-Out of Full Adder is as follows:



- Graph representation and DNA strand for the Carry-Out of Full Adder is as follows:



5. Conclusions

The field of DNA computing remains alive and promising, even as new challenges emerge. Most important among these are the uncertainty, because of the DNA chemistry, in the computational results, and the exponential increase in number of DNA molecules necessary to solve problems of interesting size. Despite these issues, definite progress has been made both in quantifying errors, and in development of new protocols for more efficient and error-tolerant DNA computing. Michael Arock et al. [7] presented a theoretical model using molecular beacons. They considered the AND gate for implementation details. In this thesis I implemented all Boolean gates. I also implemented small Combinational circuit Full Adder and Subtractor.
Following are the open subject for future work:

- Testing the concepts to implement large Boolean circuits in DNA computers.
- Extend the work by solving sequential circuits.
- Perform the complexity analysis for DNA computers.

Acknowledgements

I feel pleasure in conveying my profound thanks to my supervisor Prof. Brajesh Patel Head of Department Computer Science & Engineering, SRIT, Jabalpur, for his constant Support, valuable Guidance and Encouragement. I also thankful to all staff members of Computer Engg for their time to time supports. I am also very thankful to my family members for their cooperation and motivation.

References

[1] Russell Deaton, Max Garzon, John Rose, D. R. Franceschetti, "DNA Computing: A Review", Fundamenta Informaticae 30(23), IOS Press.
[2] Lila Kari, "DNA Computing: Arrival of Biological Mathematics", Springer-Verlag New York Volume 19, Number 2, 1997.
[3] Shrenik Shahy, Harvard University, Cambridge," APPLIED MATHEMATICS CORNER DNA Computation and Algorithm Design", 2009.
[4] Z. FRANK QIU and MI LU, Texas A&M University, "A new solution for Boolean Circuit with DNA Computer", April 10, 2000.
[5] Yasubumi Sakakibara, "DNA-based algorithms for learning Boolean Formulae", *Natural Computing* 2: 153–171, *Kluwer Academic Publisher, Japan* 2003.
[6] Michael S. Livstone, Ron Weiss and Laura F. Landweber, "Automated design and programming of a microfluidic DNA computer", Natural Computing, Springer,2006.
[7] Michael Arock , R.Ponalagusamy , and B.S.E.Zoraida , "An Efficient Algorithm for Constructing DNA Boolean Circuit ", International Journal of Computer Applications (0975 - 8887),Volume 1 – No. 22. 2010.
[8] Giuditta Franco and Vincenzo Manca, "An algorithmic analysis of DNA structure", Soft Computing, Springer-Verlag 2005.
[9] Marc Blenkiron, D. K. Arvind, and Jamie A. Davies "Design of an irreversible DNA memory element", Springer, 2007.
[10] Danny Van Noort and Laura F. Landweber,"Towards a re-programmable DNA computer", Natural Computing , Springer, 2005.
[11] Lila Kari and Kalpana Mahalingam, "Watson–Crick palindromes in DNA computing", Springer 2009.

[12] Kashif Hame, "DNA Computation Based Approach for Enhanced Computing Power", Department of Computer Science, The Islamia University of Bahawalpur, Bahawalpur, Pakistan, IJES, March 2011.

AUTHORS

Ashish Lamaniya received his B. E. degree in Computer Science and Engineering from Department of Computer Science and Engineering, from GRKIST, Jabalpur under RGPV University Bhopal (M.P.). He is currently student of Master of Engg (M.E.) at Shri Ram Institute of Technology Jabalpur. He has 3 years teaching experience of under graduate engineering students.