DNA Computing

[1]Sumit Ghulyani, [2]Varun Bajaj, [3]Niranjan Sangwan, [4]Yogesh Chandna,
[5]Ashok Kumar, [6]Rohit Kumar
[1,2,3,4,5,6]Dept. of CSE, Dronacharya College of Engineering, Gurgaon, Haryana, India

Abstract
DNA computing, also known as molecular computing, is a new loom to massively parallel calculation. The latest computer to come out of the University of Southern California isn't remarkable for its petite dimension or computational influence. It's prominent because it is made from DNA, the infinitesimal acids that exist in every cell and are accountable for all life. The DNA computer, which more narrowly resembles a biochemistry lab than a PC, was the primary non electronic device including the human mind to solve a reason dilemma with more than 1 million probable answers.
Computers in the present day all use binary codes - 1's and 0's or on's and off's. These codes are the foundation for all possible calculations a computer is capable to perform. DNA is in fact quite analogous to binary code. Each DNA strand is made up of some mishmash of A's, T's, C's and G's that act just like a computer's 1's and 0's. Furthermore, DNA copies, stores and parses information like a being hard drive and processor. Within the cell you have all the basic gear," says Adleman. "It's just a matter of shipping out the computation. In 1994, Dr. Leonard Adleman wrote the first manuscript on DNA computing. In this paper, he set up a way to solve the "Hamiltonian path predicament," which involves finding all the potential paths between a convinced numbers of vertices. It is also known as the "traveling salesman problem.

Keywords
DNA, Biochemistry, Computation

I. Introduction

DNA (deoxyribonucleic acid) is the chief genetic material in all living organisms - a molecule poised of two paired strands that are wound around each other in a double helix configuration. The strands are linked by base pairs that look like rungs in a stepladder. Each base will pair with only one other: adenine (A) pairs with thymine (T), guanine (G) pairs with cytosine (C). The progression of each single strand can therefore be deduced by the identity of its cohort.
Genes are sections of DNA that code for a defined biochemical function, usually the fabrication of a protein. The DNA of an organism may enclose anywhere from a dozen genes, as in a virus, to tens of thousands of genes in higher organisms like humans. The makeup of a protein determines its task. The succession of bases in a given gene determines the structure of a protein. Thus the genetic code determines what proteins an organism can make and what those proteins can do. It is estimated that only 1-3% of the DNA in our cells codes for genes; the rest may be used as a decoy to absorb mutations that could otherwise damage vital genes.
mRNA (Messenger RNA) is used to relay information from a gene to the protein synthesis apparatus in cells. mRNA is made by copying the series of a gene, with one subtle difference: thymine (T) in DNA is substituted by uracil (U) in mRNA. This allows cells to differentiate mRNA from DNA so that mRNA can be selectively tainted without destroying DNA. The DNA-o-gram initiator simplifies this step by taking mRNA out of the equation.
The genetic code is the language used by living cells to convert information found in DNA into information needed to make proteins. A protein's structure, and therefore function, is determined by the sequence of amino acid subunits. The amino acid sequence of a protein is determined by the sequence of the gene programming that protein. The "words" of the genetic code are called codons. Each codon consists of three adjoining bases in an mRNA molecule. Using combinations of A, U, C and G, there can be sixty four diverse three-base codons. There are only twenty amino acids that require to be coded for by these sixty four codons. This glut of codons is known as the redundancy of the genetic code. By allowing more than one codon to specify each amino acid, mutations can occur in the sequence of a gene without upsetting the resulting protein. The DNA-o-gram generator uses the genetic code to specify letters of the alphabet instead of coding for proteins.

II. DNA Computing

A DNA computer is a compilation of DNA strands that have been specially selected to aid in the hunt of solutions for some tribulations. DNA computing results in parallelism, which means that when enough DNA information is given, huge problems can be solved by invoking a corresponding search.
1994, Leonard M. Adleman solved an unremarkable computational crisis with a remarkable technique. It was a problem that a person could solve it in a few moments or an average desktop engine could solve in the blink of an eye. It took Adleman, however, seven days to find a elucidation. This work was exceptional, because he solved the problem with DNA. It was a milestone demonstration of computing on the molecular level.
The type of problem that Adleman solved is a famous one. It's formally known as a directed Hamiltonian Path (HP) problem, but is more popularly recognized as a variant of the so-called "traveling salesman problem." In Adleman's version of the traveling salesman problem, or "TSP" for short, a supposed salesman tries to find a itinerary through a set of cities so that he visits each city only once. As the number of cities increases, the problem becomes more difficult until its solution is beyond logical analysis altogether, at which point it requires swine force search methods. TSPs with a large number of cities quickly become computationally classy, making them impractical to solve on even the latest super-computer. Adleman's manifestation only involves seven cities, making it in some sense a trivial problem that can easily be solved by examination. Nevertheless, his work is significant for a number of reasons.

- It illustrates the possibilities of using DNA to solve a class of troubles that is difficult or impossible to crack using traditional computing methods.
- It's an example of computation at a molecular level, potentially a size limit that may never be reached by the semiconductor business.
- It demonstrates unique aspects of DNA as a data structure
- It demonstrates that computing with DNA can work in a massively analogous fashion.

A. The Data Density of DNA is Impressive

Just like a string of binary data is encoded with ones and zeros, a strand of DNA is encoded with four bases, represented by the

letters A, T, C, and G. The bases (also known as nucleotides) are spaced every 0.35 nanometers along the DNA molecule, giving DNA an remarkable data density of nearly 18 Mbits per inch. In two dimensions, if you presume one base per square nanometer, the data solidity is over one million Gbits per square inch. Judge against this to the data density of a typical high show hard drive, which is about 7 Gbits per square inch—a factor of over 100,000 smaller.

B. DNA in Its Double Stranded Nature

The bases A and T, and C and G, can bind together, forming base pairs. Therefore every DNA sequence has a natural accompaniment. For example if sequence S is ATTACGTCG, its complement, S', is TAATGCAGC. Both S and S' will come collectively to form double stranded DNA. This complement makes DNA a unique data structure for computation and can be subjugated in many ways. Error correction is one example. Errors in DNA happen due to many factors. Occasionally, DNA enzymes merely make mistakes, cutting where they shouldn't, or inserting a T for a G. DNA can also be dented by thermal force and UV energy from the sun. If the error occurs in one of the strands of double stranded DNA, repair enzymes can restore the proper DNA sequence by using the complement strand as a reference. In this sense, double stranded DNA is similar to a RAID 1 array, where data is mirrored on two drives, allowing data to be recovered from the subsequent drive if errors occur on the first. In biological systems, this flair for error rectification means that the error rate can be quite low. For example, in DNA replication, there is one blunder for every 10^9 copied bases or in other words an error rate of 10^-9.

C. Operations in Parallel

In the cell, DNA is tailored biochemical by a variety of enzymes, which are tiny protein machines that read and process DNA according to nature's design. There is a wide range and number of these "operational" proteins, which maneuver DNA on the molecular level. For example, there are enzymes that cut DNA and enzymes that stick it back together. Other enzymes function as copiers and others as repair units. Molecular biology, Biochemistry, and Biotechnology have developed techniques that allow us to execute many of these cellular functions in the test-tube.

It's this cellular machinery, along with some mock chemistry, that makes up the palette of operations obtainable for computation. Just like a CPU has a basic suite of operations like addition, bit-shifting, logical operators (AND, OR, NOT NOR), etc. that permit it to perform even the most complex calculations, DNA has cutting, copying, pasting, repairing, and many others. And note that in the test tube; enzymes do not function sequentially, functioning on one DNA at a time. Rather, many copies of the enzyme can work on many DNA molecules simultaneously. This is the power of DNA computing, that it can work in a massively parallel trend.

III. The Adleman Experiment

There is no better way to comprehend how something works than by going through an example step by step. So let's solve our own directed Hamiltonian Path problem, using the DNA methods demonstrated by Adleman. The concepts are the same but the example has been cut down to make it easier to chase and present.

Suppose that I live in LA, and need to visit four cities: Dallas, Chicago, Miami, and New York, with New York 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 voyage from L.A. to Chicago, but no flight from Miami to Chicago). What should my tour be if I want to visit each city only once? [1].

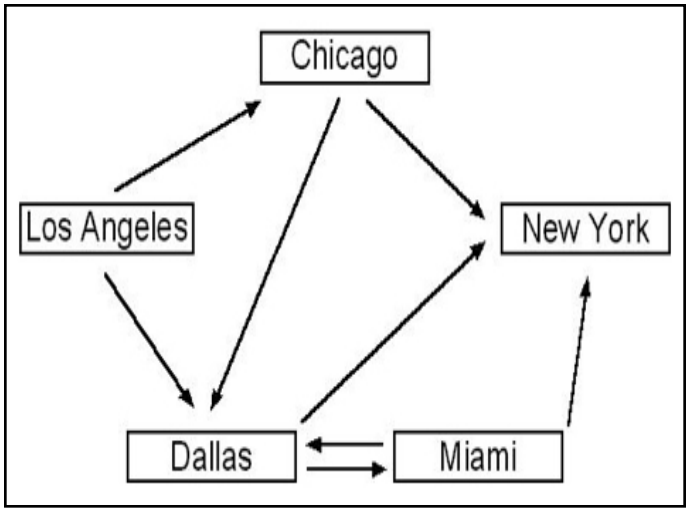

Fig. 1: The Adleman Experiment

It should take you only a moment to see that there is only one route. Starting from L.A. you need to fly to Chicago, Dallas, Miami and then to N.Y. Any other choice of cities will force you to miss a intention, visit a city twice, or not make it to N.Y. For this example you obviously don't need the help of a computer to find a solution. For six, seven, or even eight cities, the problem is still manageable. However, as the number of cities increases, the problem quickly gets out of hand. Assuming a random division of connecting routes, the number of itineraries you need to check increases exponentially. Pretty soon you will run out of pen and paper listing all the possible routes, and it becomes a problem for a computer.....or perhaps DNA. The method Adleman used to solve this difficulty is basically the shotgun loom mentioned previously. He first generated all the possible itineraries and then selected the correct itinerary. This is the pro 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 assortment process is massively parallel. Specifically, the technique based on Adleman's experiment would be as follows:

- Spawn all possible routes.
- Select itineraries that begin with the proper city and end with the final city.
- Select itineraries with the exact number of cities.
- Select itineraries that contain each city only once.

All of the above steps can be proficient with standard molecular biology techniques.

A. Generate all Possible Routes

1. Strategy

Encode city names in short DNA sequences. Encode itineraries by connecting the city sequences for which routes exist.

DNA can simply be treated as a string of data. For example, each city can be represented by a "word" of six bases:

Los Angeles	GCTACG
Chicago	CTAGTA
Dallas	TCGTAC
Miami	CTACGG
New York	ATGCCG

The entire itinerary can be encoded by simply stringing together these DNA sequences that represent specific cities. For example, the route from
L.A -> Chicago -> Dallas -> Miami -> New York would simply be
GCTACGCTAGTATCGTACCTACGGATGCCG, or unvaryingly it could be represented in double stranded outline with its balance sequence.
So how do we generate this? Synthesizing short single stranded DNA is now a routine process, so encoding the city names is straightforward. The molecules can be made by a machine called a DNA synthesizer or even routine ordered from a third party. Itineraries can then be produced from the city encodings by linking them collectively in proper order. To accomplish this you can take help of the fact that DNA hybridizes with its complimentary sequence. For example, you can encode the routes between cities by encoding the compliment of the second half (last three letters) of the departure city and the first half (first three letters) of the arrival city. For example the path between Miami (CTACGG) and NY (ATGCCG) can be made by taking the next half of the coding for Miami (CGG) and the first half of the coding for NY (ATG).
This gives CGGATG. By taking the complement of this you get, GCCTAC, which not only uniquely represents the route from Miami to NY, but will tie the DNA representing Miami and NY by hybridizing itself to the second half of the cipher representing Miami (...CGG) and the first half of the code representing NY (ATG...). For example:
Random itineraries can be made by amalgamation city encodings with the route encodings. Finally, the DNA strands can be connected together by an enzyme called ligase. What we are left with are strands of DNA representing itineraries with a random number of cities and random set of routes. For example:
We can be convinced that we have all possible combinations including the correct one by using an excess of DNA encodings, say 10^13 copies of each city and each route amid cities. Remember DNA is a highly compact data layout, so numbers are on our side.

B. Select Itineraries That Start and End with the Correct Cities

1. Strategy

Selectively copy and amplify only the section of the DNA that starts with LA and ends with NY by using the Polymerase Chain Reaction.
"After what we did in Part A, we now have a test tube full of various lengths of DNA that encode possible routes between cities. What we want are routes that start with LA and end with NY. To accomplish this we can use a technique called Polymerase Chain Reaction (PCR), .Polymerase chain reaction (PCR), is a general method of creating copies of specific fragments of DNA. PCR swiftly amplifies a single DNA molecule into many billions of molecules. PCR exploits the remarkable natural function of the enzymes known as polymerases. These enzymes are there in all living things, and their trade is to copy genetic material (and also proofread and correct the copies). Sometimes referred to as "molecular photocopying," PCR can typify, analyze, and synthesize any specific part of DNA or RNA.
It works even on extremely complex mixtures, seeking out, identifying, and duplicating a particular bit of genetic material from blood, hair, or tissue specimens, from microbes, animals, or plants, some of them many thousands-or possibly even millions-of years old.
PCR requires a stencil molecule-the DNA or RNA you want to copy-and two primer molecules to get the copying process on track. The primers are short chains of the four different chemical components that make up any strand of genetic material. These four components are like bricks or building blocks that are used to erect genetic molecules; in the lab they are called nucleotides or bases [2].
DNA itself is a chain of nucleotides. Under most circumstances, DNA is double-stranded, consisting of two such nucleotide chains that wind around each other in the prominent shape known as the double helix. Primers are single-stranded. They consist of a string of nucleotides in a specific order that will, under the right conditions, bind to a specific complementary sequence of nucleotides in another piece of single-stranded RNA or DNA.
For PCR, primers must be duplicates of nucleotide sequences on either side of the piece of DNA of interest, which means that the exact order of the primers' nucleotides must already be known. These contiguous sequences can be constructed in the lab, or purchased from commercial suppliers. There are three basic steps in PCR. First, the target genetic material must be denatured-that is, the strands of its helix must be unwound and separated-by heating to 90-96°C. The following step is hybridization or annealing, in which the primers truss to their complementary bases on the now single-stranded DNA. The third is DNA synthesis by a polymerase. Starting from the primer, the polymerase can interpret a template strand and match it with complementary nucleotides very quickly. The result is two new helixes in place of the first, each composed of one of the original strands plus its newly assembled complementary strand [3].
All PCR really requires in the way of equipment is a reaction tube, reagents, and a cause of heat. But different temperatures are optimal for each of the three steps, so machines now control these temperature variations automatically. To get more of the DNA you want, just repeat the process, commencement by denaturing the DNA you've already made. The amounts will double every time. So to selectively intensify the itineraries that start and stop with our cities of interest, we use primers that are complimentary to LA and NY. What we end up with after PCR is a test tube full of double stranded DNA of various lengths, encoding itineraries that start with LA and end with NY.

C. Select Itineraries That Contain the Correct Number of Cities

1. Strategy

Sort the DNA by length and select the DNA whose length corresponds to 5 cities.
Our test tube is now crammed with DNA encoded itineraries that start with LA and end with NY, where the number of cities in between LA and NY varies. We now want to opt for those itineraries that are five cities long. To accomplish this we can use a technique called Gel Electrophoresis, which is a common procedure used to resolve the size of DNA. The basic principle behind Gel Electrophoresis is to force DNA through a gel matrix by using an electric field. DNA is a negatively charged molecule under most conditions, so if placed in an electric field it will be fascinated to the positive potential. However since the charge density of DNA is constant (charge per length) long pieces of DNA move as fast as short pieces when perched in a fluid. This is why you use a gel matrix.

The gel is made up of a polymer that forms a meshwork of linked strands. The DNA now is forced to cord its way through the tiny spaces between these strands, which slows down the DNA at different rates depending on its length. What we typically end up with after running a gel is a series of DNA bands, with each band corresponding to a certain length. We can then simply cut out the band of curiosity to isolate DNA of a specific length. Since we known that each city is encoded with 6 base pairs of DNA, knowing the length of the itinerary gives us the number of cities. In this case we would segregate the DNA that was 30 base pairs long (5 cities times 6 base pairs).

D. Select Itineraries that have a Complete Set of Cities

1. Strategy

Successively filter the DNA molecules by city, one city at a time. Since the DNA we start with contains five cities, we will be left with strands that encode each city once.

DNA containing a explicit sequence can be purified from a sample of mixed DNA by a technique called affinity purification. This is accomplished by attaching the compliment of the sequence in question to a substrate like a magnetic bead. The beads are then mixed with the DNA. DNA, which contains the sequence you're after then hybridizes with the complement sequence on the beads. These beads can then be retrieved and the DNA secluded.

So we now purify our refines five times, using a different city complement for each run. For example, for the first run we use L.A.'-beads (where the 'indicates compliment strand) to fish out DNA sequences which contain the encoding for L.A. (which should be the entire DNA because of step 3), the next run we use Dallas'-beads, and then Chicago'-beads, Miami'-beads, and finally NY'-beads. The order isn't vital. If an itinerary is missing a city, then it will not be "fished out" during one of the runs and will be removed from the contender pool. What we are left with are the itineraries that start in LA, visit each city once, and end in NY. This is unerringly what we are looking for. If the answer exists we would retrieve it at this step.

E. Reading out the Answer

One possible way to find the result would be to simply sequence the DNA strands. However, since we already have the sequence of the city encodings we can use an exchange method called graduated PCR. Here we do a series of PCR amplifications using the primer corresponding to L.A., with a different primer for each city in succession. By measuring the various lengths of DNA for each PCR product we can piece collectively the final sequence of cities in our itinerary. For example, we know that the DNA itinerary starts with LA and is 30 base pairs long, so if the PCR product for the LA and Dallas primers was 24 base pairs long, you know Dallas is the fourth city in the circuit (24 divided by 6). Finally, if we were cautious in our DNA manipulations the only DNA left in our test tube should be DNA itinerary encoding LA, Chicago, Miami, Dallas, and NY. So if the progression of primers used is LA & Chicago, LA & Miami, LA & Dallas, and LA & NY, then we would get PCR products with lengths 12, 18, 24, and 30 base pairs.

IV. DNA vs. Silicon

DNA, with its exclusive data structure and ability to complete many parallel operations, allows you to look at a computational dilemma from a different point of view. Transistor-based computers typically handle operations in a sequential manner. Of course there are multi-processor computers, and contemporary CPUs incorporate some analogous processing, but in general, in the basic von Neumann architecture computer, instructions are handled sequentially. A von Neumann machine, which is what all modern CPUs are, basically repeats the equivalent "fetch and execute cycle" over and over again; it fetches an instruction and the apposite data from main memory, and it executes the instruction. It does these many times in a row, really, really prompt. The great Richard Feynman, in his Lectures on Computation, summed up von Neumann computers by aphorism, "the inside of a computer is as dumb as hell, but it goes like crazy!" DNA computers, however, are non-von Neuman, stochastic machines that advance computation in an unusual way from ordinary computers for the purpose of solving a different class of problems.

Typically, escalating performance of silicon computing means faster clock cycles (and larger data paths), where the accent is on the alacrity of the CPU and not on the size of the memory. For example, will doubling-up the clock momentum or copying your RAM give you healthier performance? For DNA computing, though, the power comes from the recall capacity and parallel processing. If forced to act sequentially, DNA loses its appeal. For instance, let's look at the read and carve charge of DNA. In bacteria, DNA can be replicated at a rate of about 500 base pairs a second. Biologically this is pretty fast (10 times nearer than human cells) and considering the squat error rates, an impressive triumph. But this is only 1000 bits/sec, which is a snail's lick when compared to the data throughput of an average hard drive.

But look what happens if you consent to many copies of the replication enzymes to toil on DNA in parallel. First of all, the replication enzymes can initiate on the second replicated strand of DNA even prior to they're finished copying the first one. So by now the data rate jumps to 2000 bits/sec. But look what happens after each imitation is finished - the number of DNA strands increases exponentially (2^n after n iterations). With each supplementary strand, the data rate increases by 1000 bits/sec. So after 10 iterations, the DNA is being replicated at a rate of about 1Mbit/sec; after 30 iterations it increases to 1000 Gbits/sec. This is afar the sustained data rates of the fastest firm drives [4].

Now let's consider how you would solve a nontrivial paradigm of the traveling salesman problem (# of cities > 10) with silicon vs. DNA. With a von Neumann computer, one naive scheme would be to set up a search tree, measure each absolute branch sequentially, and keep the shortest one. Improvements could be made with enhanced search algorithms, such as pruning the search tree when one of the branches you are measuring is already longer than the best aspirant. A method you certainly would not use would be to first spawn all possible paths and then search the entire list. Why? Well, deem that the entire inventory of routes for a 20 city problem could hypothetically take 45 million GBytes of memory (18! routes with 7 byte words)! Also for a 100 MIPS computer, it would take two years just to breed all paths (assuming one instruction phase to engender each city in every trail). However, using DNA computing, this process becomes realistic! 10^15 is just a nanometer mole of substance, a relatively small number for biochemistry [5]. Also, routes no longer have to be searched through successively. Operations can be completed all in parallel.

V. The Future

Premature computers such as ENIAC filled intact rooms, and had to be programmed by punch cards. Since that era, computers

have since become much smaller and easier to use. It is possible that DNA computers will become more common for solving very multifaceted problems, and just as PCR and DNA sequencing were once handbook tasks, DNA computers may also become automated. In count to the express benefits of using DNA computers for performing complex computations, some of the operations of DNA computers already have, and perceivably more will be used in molecular and biochemical study.

Despite its promise, the long-term diagnosis for DNA computing remains uncertain. The opinion of full-scale DNA computing systems replacing silicon-based computers anytime presently, if ever, is remote. There are still momentous limitations to conquer. For the projected future - and as its forge Adleman suggests - DNA computing will likely center on small-scale applications rather than the edifice of full-blown computers [6].

The university team created a crude molecular computer "chip" made of a diminutive glass plate covered with a thin stratum of gold strands of DNA was coded to symbolize solutions to a computational problem with 16 possible answers. Then, enzymes were applied to the gold slide to strip out the entire DNA with the erroneous answers and, and thus, solving the estimate. "It opens up the likelihood of ultrahigh-capacity storage and massively analogous searches," explains Robert Corn, a lecturer of chemistry and a affiliate of the research team. A DNA computer the size of a fifty pence piece, for exemplar, could hold up to 10 terabytes of data, far beyond the capacity of any computer storage medium on hand today.

The study on DNA computers is ongoing still. All over, research teams are concentrating their pains in order to put this novel nanotechnology to good use. And even though Adleman's DNA computer would have a hard time computing two 100-digit integers - an easy task for a supercomputer - its ability to solve complex problems is unmatched. As this new nanotechnology continues to progress, we might yet be surprised again. The DNA based system of computing has had millions of years to evolve, while the man-made systems have only been around for a small portion of that time. The outlook of DNA computing has yet to be decided. Anne Condon, a computer scientist on the Wisconsin team, likens compares current DNA computing to that of ENIAC computers. Built in 1946, ENIAC computers used punch cards and closets full of vacuum tubes to solve plain arithmetic problems.

"It's doable that we could utilize DNA computers to manage compound and biological systems in a way that's analogous to the mode we use electronic computers to run electrical and perfunctory systems," Adleman said.

VI. Conclusion

So will DNA ever be used to decipher a wandering salesman problem with a senior number of cities than can be done with traditional computers? Well, bearing in mind that the record is a monstrous 13,509 cities, it certainly will not be done with the modus operandi described above. It took this only three months, using three Digital Alpha Server 4100s (a total of 12 processors) and a huddle of 32 Pentium-II PCs. The key was possible not because of brute force computing influence, but because they used some very proficient branching rules. This first demonstration of DNA computing used a rather crude algorithm, but as the formalism of DNA computing becomes refined, new algorithms perhaps will one day agree to DNA to overhaul conventional computation and set a new documentation. Thus far, Adleman has only experienced his DNA model with six vertices and is tentative as to how to proceed with paths of more than six vertices. But as far as speed is concerned, DNA evidently wins.

The best ever supercomputers today execute about 1,012 operations per microsecond, while the DNA models perform 1,000 times faster than the fastest super computer A clearer picture of this - and probably one that we can narrate to better - is that of the archetypal desktop computer. Our desktops put to death 106 operations per microsecond, which is a thousand million times slower than the DNA.DNA computing will likely spotlight on small-scale applications rather than the construction of full-blown computers.

References

[1] Adleman, L.,"Molecular computation of results to combinational problems. Science 266, pp. 1021-1024, 1994.

[2] Lipton, R. J.,"Speeding up computations via molecular biology".

[3] Boneh, D., Lipton, R. J.,"Making DNA computers error resistant".

[4] Kari, L.,"DNA computing: the arrival of biological mathematics", 1997.

[5] Adleman, L.,"On constructing a molecular computer", 1995.

[6] Paun, G.,"Computing with Bio-Molecules", Springer-Verlag, 1998.

Sumit Ghulyani is presently a final year student pursuing degree in Bachelor of Technology in the field of Computer Science Engineering from Dronacharya College of Engineering, Gurgaon, India. Other than being the campus ambassador for more than a couple of organisations, he has been one of the soul limbs in successfully coordinating & hosting technical events with Microsoft and HRs of various other technical giants. Authored many significant papers, other fields of his research interest include Brain Computer Interface, Windows Operating Systems, Apple MAC OS, Android Operating Systems, Wireless Power Transmission, Artificial Intelligence & Virtual Reality.