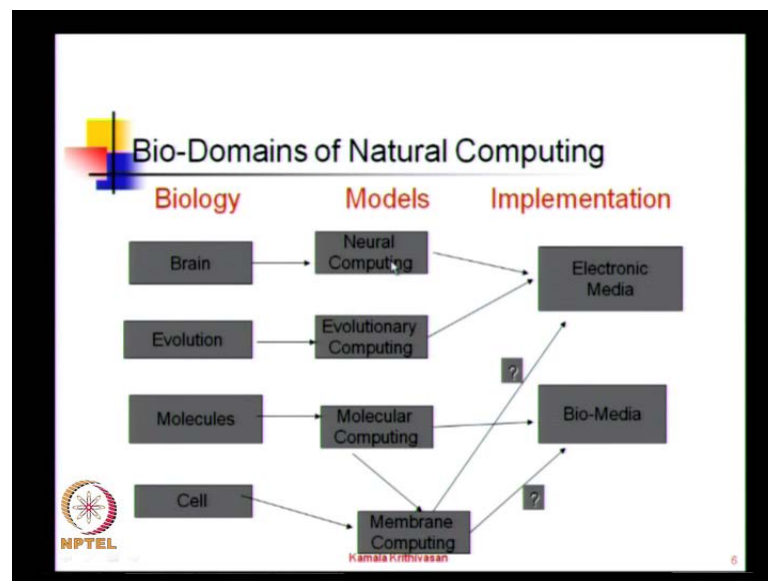


Theory of Computation
Prof. Kamala Krithivasan
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture No. # 41
DNA Computing

(Refer Slide Time: 00:23)



So, we shall consider a new model of computing namely, DNA computing; it is a new paradigm, and it is one of the models of natural computing. So, if you look into the different branches of natural computing, you can divide them like this. Models inspired by nature that is Bio-Inspired computing under this comes Neural Computing, you know that nowadays neural networks play a very important role in the field of computer science. And they have come into existence, because of the simulation of brain; in neural computing, you try to simulate the way the brain works; and the neural networks are implemented with the electronic silicon computer. So, the implementation is through the electronic media.

And you also have what is known as an evolutionary computing, genetic algorithms. They try to solve an algorithm making use of representing an instance using a bit string. And then they perform the operations of cross over mutation selection etcetera, which

happens in the actual evolution of humans or any other organism in nature. And because they simulate the evolution, it is called the evolutionary computing and genetic algorithms come under this. Genetic algorithms are mostly heuristic algorithms and even though they are heuristic algorithms.

In most cases, they give very good solutions to some problems because they converge, even though the convergence cannot be properly proved. In most of the examples where it is considered, it is found that by taking the model in a suitable manner and representing the strings in a suitable manner, the convergence can be achieved. Even for network algorithms like routing etcetera, evolutionary computing has been used and it is a very good area of research nowadays.

You also have what is known as swarm intelligence and colony paradigms and so on which are biologically inspired. In contrast to that you also have some models which simulate nature and show how nature is being developed like fractal geometry and cellular automata. You might have heard about the game of life and so on how it is done with cellular automata. That is the second one; the third one is actually computing with molecules or computing with natural objects that come under molecular computing. The actual thing is the molecules and how they behave and interact with each other, giving solutions where problems. And that is called molecular computing, DNA computing, peptide computing they all come under this.

And the implementation is really using the DNA strands or the peptides and the molecules. So, the implementation is through bio-media in 1998, the paradigm membrane computing was proposed because it tried to simulate the behaviour of what happens in a cell, the proteins the enzymes the membrane structure? They how they develop? They pass through the membranes changing from one form to another and so on. And initially the theoretical model was developed and the first paper in this area was published in 2000. At that time it was thought that there are two possibilities, one it could be implemented using the silicon computer like evolutionary algorithms and so on.

The other one is you may actually use membranes or cells and try to simulate the algorithms this was the question in 2000. And in 2010 now, this looks that it is not possible at all how to make them make the algorithms be simulated by actual cells. Whereas, this looks more promising that is, this may develop as a bio-inspired model of

computing, like evolutionary computing or neural computing and will be implemented in the electronic media only.

(Refer Slide Time: 05:06)

The slide features a diagram at the top with two boxes: "Biological Mathematics" on the left and "Mathematical Biology" on the right. A square connects them, with an arrow labeled "Mathematics" pointing from the right box to the left box, and an arrow labeled "Biology" pointing from the left box to the right box.

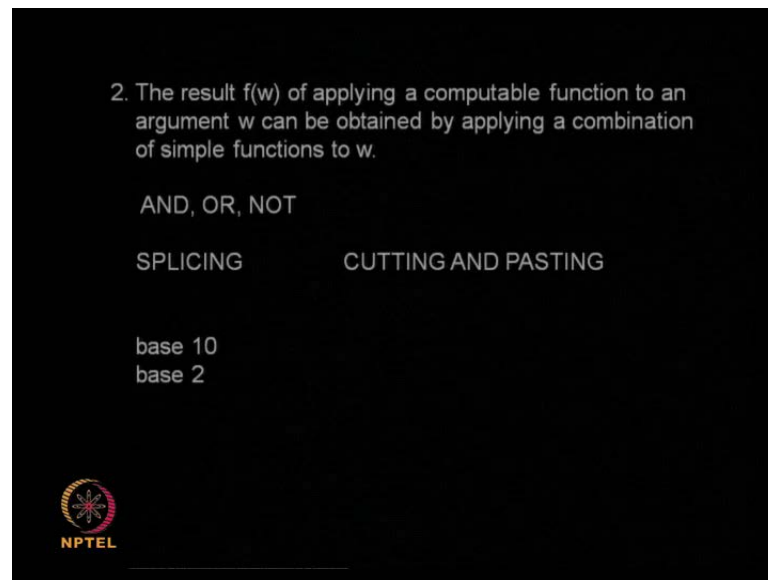
The idea behind biological mathematics is the simple observation that the following two processes one biological and one mathematical are analogous

1. The very complex structure of a living being is the result of applying simple observations (copying, splicing, etc) to initial information encoded in a DNA sequence.

The NPTEL logo is located in the bottom left corner of the slide.

Molecular computing or DNA computing is implemented by bio-media and what is the motivation for this? You talk about mathematical biology and biological mathematics. What is this? In one way mathematics tries to help biology and in another way biology tells tries to help mathematics or computing. The reason or the comparison between these two models is that, the biological mathematics is simple observation. That the following two processes are very much analogous what are the processes? The very complex structure of a living being is the result of applying simple operations, observations like copying, splicing etcetera. To initial information encoded in a DNA sequence, simple operations continuously applied gives rise to many things.

(Refer Slide Time: 06:16)



And that helps the behaviour of the DNA strands. DNA evolves and the basic operations are cutting and splicing, cutting and pasting, splicing etcetera. In mathematics you can look at the function computed anything should be computed, you can look at it as a function f computing something, w is the input and $f w$ is the output. The result $f w$ of applying a computable function to an argument w the result given is $f w$, it can be obtained by applying a combination of simple functions to w . Even though f may be a very complex function, it can be obtained from simple functions by continuous applications; this is the definition of partial recursive functions.

Any function any computable function; you can define making use of the initial functions and then using the operations of composition primitive recursion and minimization. So, these two processes are very much similar and the similarity between them tells the connection between mathematics and biology. Mathematics has always help biology in forming hypothesis, in stochastic processes is one of the thing which helped biology a lot. And you try to form hypothesis giving the theoretical background and verify the hypothesis. The other question is can biology inspire mathematics or can biology help mathematics and that is answered by DNA computing, actually the DNA strands or biological material is helpful in doing some computation, that is what we are going to see.

And one way to look at it is any Boolean circuit can be formed from the three gates and or not in fact, two of them is enough. So, basically if you have and or not gate any Boolean function can be implemented similarly, some basic operation will help the DNA strand to evolve in any manner you want this initially may look a little bit puzzling and you may not accept it. And this was a situation earlier about 40 years back when somebody said 1 0 1 0 it meant 1010 and it did not mean number 10. So, the binary representation, people were aware of but it was not very familiar, whenever some number is stated they really thought it is base 10. But now, when somebody says 1 0 0 1 immediately the mind will grasp it as 1001 or nine number 9 with the base 2.

Base 2 has become part and parcel of our life with the invention of computers, but it was not so, about 40 years back. So, that is what I want to say, that is today DNA computers may look something new, something not acceptable, but, after 20 years it may become part and parcel of our life this is a current situation. And in DNA computing you want the whole computation to be done like this, with DNA strands using bio molecules using test tubes and so on.

But at the current stage this is not completely possible involving lot of cost and time, we want to do everything in a very quick manner using less cost. And that is not possible today, but it may become possible after some time, but how much it is going to be possible is a question. It still a question, but finally, we may end up something like this because the operations have to be performed manually. It may take a lot of time, you may use some DNA strands, but then the operations may be performed by a robot, which is really a silicon computer. So, you may have a mixture of both electronic media and bio-media and end up with a situation like this, where things can be done in a much efficient and in a fast way.


(Refer Slide Time: 11:04)

DNA Deoxyribonucleohides
Deoxyribo nuclic acid

A sugar phospahte group and consists of nitrogenous base

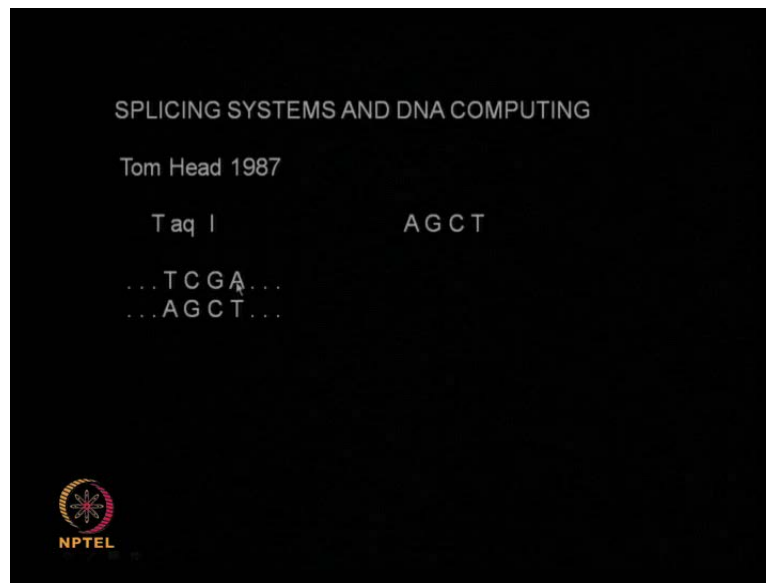
Purines	adenine	A
	guanine	G
Pyrimidines	thymine	T
	cytosine	C

RNA – ribonucleic acid



So, what is DNA computing? The first paper or the first work was done by Adelman in 1994. He tried to find out whether a given graph has a Hamiltonian path and he developed a new algorithm for that, using DNA stands and experimentally verified that. And that was a breakthrough result and after that a lot of experiments and lot of research has going on in this area. Now, coming back what is a DNA? DNA is Deoxyribonucleohides and it has got a sugar part of it a phosphate group and consists of a nitrogenous base, it can be divided into two groups the purines and pyrimidines. The purines are adenine and guanine and they are represented by the letters A and G, the pyrimidines are thymine and cytosine they are represented by the letters T and C. RNA is ribonucleic acid.

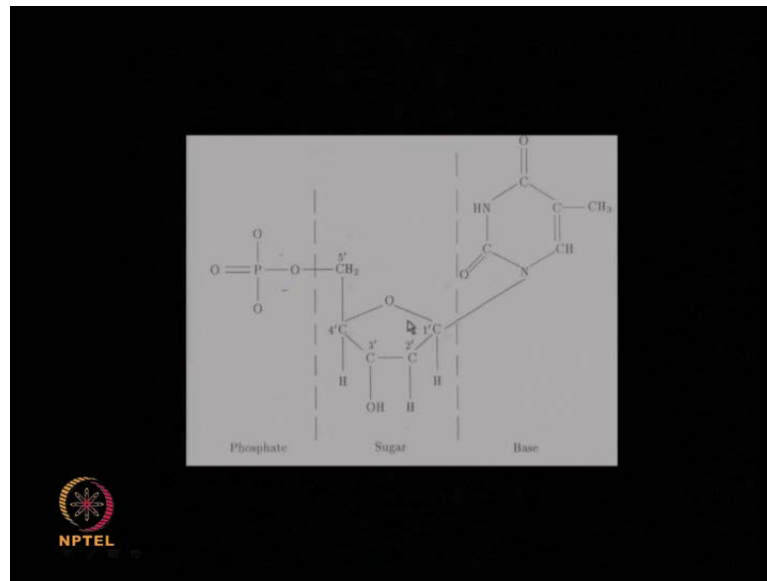
(Refer Slide Time: 12:14)



Now, this splicing system was defined that is the DNA can be used to model some behaviour. Recombination of the DNA's can be modelled by using what is known as a splicing system? This sort of a theoretical work was done by tom head 1987 itself, but in 1994, when Adelman published his paper, then only lot of interest was created in this area. And this is one of the reasons for the development of formal languages and automata theory new models of generating languages have been defined like splicing systems, sticker systems and so on.

So, what is a DNA? DNA is a double stranded, it has a helical structure and it double stranded, this was found by Watson and Crick for which they were awarded the Nobel Prize. And T always combines with A and C always combines with G, they are called complimentary pairs. So, A always pairs with T G always pairs with C and there is a double hydrogen bond between them A and T and a triple hydrogen bond between C and G.

(Refer Slide Time: 13:32)



So, the structure of a DNA is something like this, you have a sugar part and you have a base part, this will be different A G C T and the phosphate part. And the carbon molecules are named as 1 2 3 4 5. So, this is called 1 2 3 4 and 5.

(Refer Slide Time: 13:57)

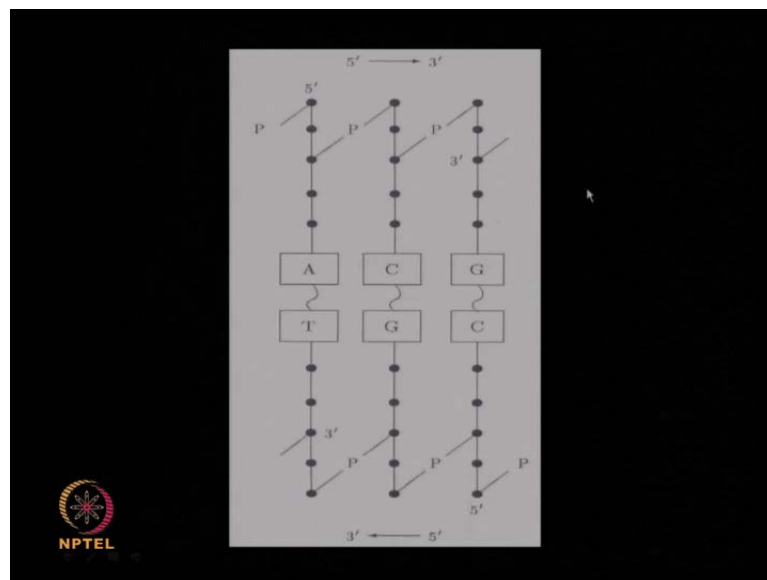


So, you can see that A always will combine with T only, T has a structure like this, to represent in a pictorial form and A can only combine with, A cannot combine with C or G and C and G only can combine. So, you can see that suppose, I have one single strand A G C A, then I can put this T G here and it will become like this, properly matching and

here for G I should have C and for A I should have T and so, this can be brought here and then I am having a sticky end here. So, it can be combined, this is a double strand and there is a sticky end at this portion.

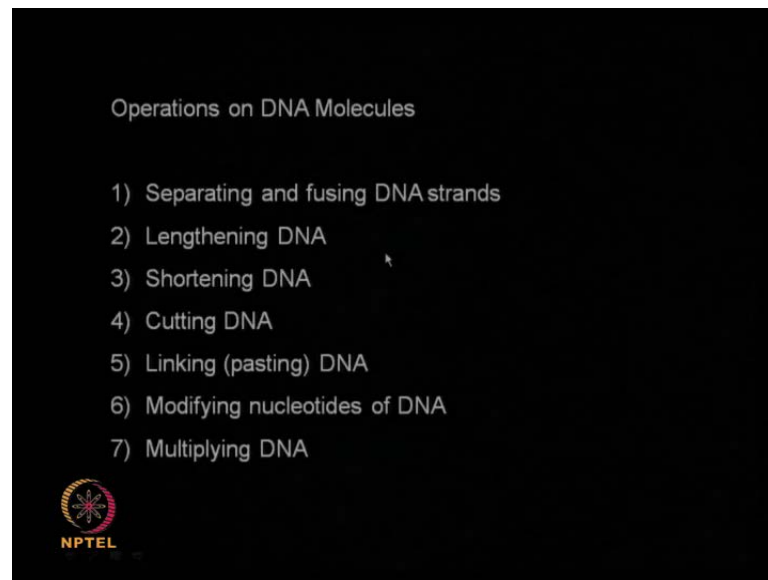
This helps to form the second strand if you know the first strand suppose, I have a lengthy DNA single strand and then the second strand I do not know, but I have several pieces of second strands and which one will match with this. If anything is allowed it will be very difficult to match whereas, the Watson-crick complementarily helps us to form the second strand in a very convenient manner. And this is the basis this and immense parallelism is the basis for all DNA algorithms. So, with that you can form the double strand in a very convenient manner and you see how easily this has been formed.

(Refer Slide Time: 15:32)



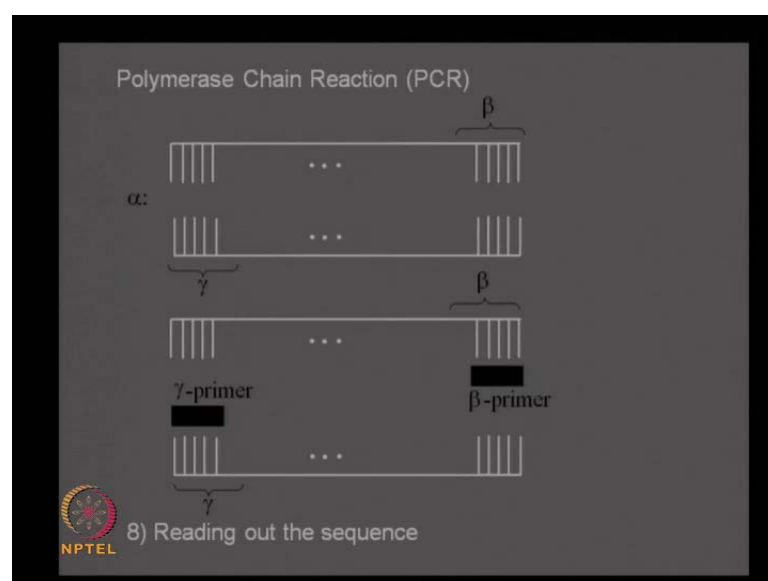
Also the double strand is represented like this, one strand is represented as 5 dash to 3 dash end, another one is represented as 3 dash to 5 dash end. So, you have the 5 carbon atoms, the last one here with the phosphate structure it will combine with the third one here and so on. Like that these A C G, they will be combined by bonds like this and here again 5 will combine with 3 and so on. So, this one is called denoted as 3 dash to 5 dash end, this one is denoted as 5 dash to 3 dash end. And these always pair together there is a double hydrogen bond between A and T and a triple hydrogen bond between C and G.

(Refer Slide Time: 16:26)



So, the operations which is usually done on DNA molecules, they can be classified like this separating and fusing of DNA strands. DNA's are double stranded by heating they may become single stranded and by cooling and adding ligase they may join and form a double strand. Single strands can join and form a double strand lengthening of a DNA, given DNA strand you keep on increasing the length shortening cutting of a DNA using a proper enzyme at the restriction site. And 2 DNA can be linked together using ligase and nucleotides can be modified a little bit, then you can increase the number of DNA strand multiply them. So, that helps in identifying the proper strand.

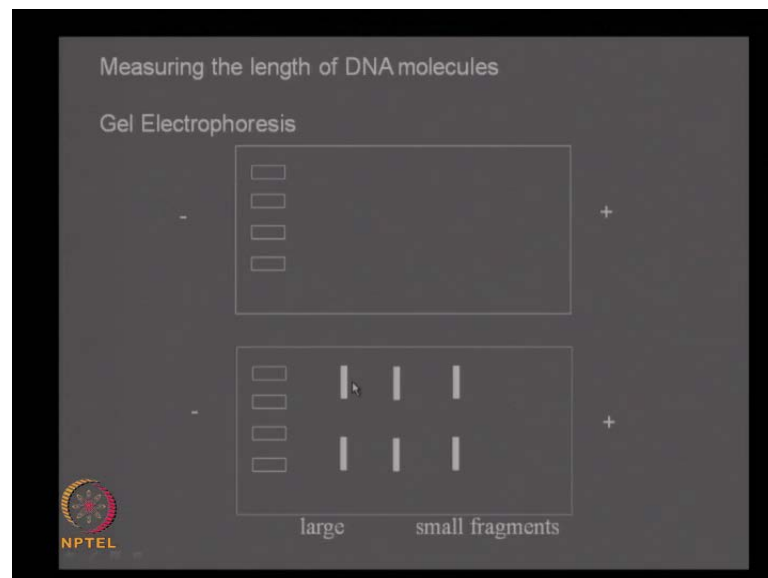
(Refer Slide Time: 17:31)



The multiplication of DNA strands are it is called amplification, it is done using what is known as polymerase chain reaction? A double stranded DNA is given like this, to get four strands from this, is the single double strand from this you can get two double strands in the following manner, you heat it to a proper temperature. So, the double strand becomes single strands and you add what is known as a beta primer and a gamma primer? What is beta primer? Is the compliment of the few nucleotides here, when you add and then use polymerase chain reaction, this will keep on warming the second strand and the whole thing will become a double strand.

Similarly, adding a gamma primer here, that is the compliment of this portion here and this will be lengthened and you form another double strand. So, from one double strand DNA you can get two double stranded DNA, if you repeat the process you will get 4 8 16 and so on. So, after some time the number of double strands will increase and this is very essential in some operations, we shall see that and reading out the sequence there is a specific way you can read out the sequence of the DNA, we will not go into that Sanger method and other methods are available for that.

(Refer Slide Time: 19:13)

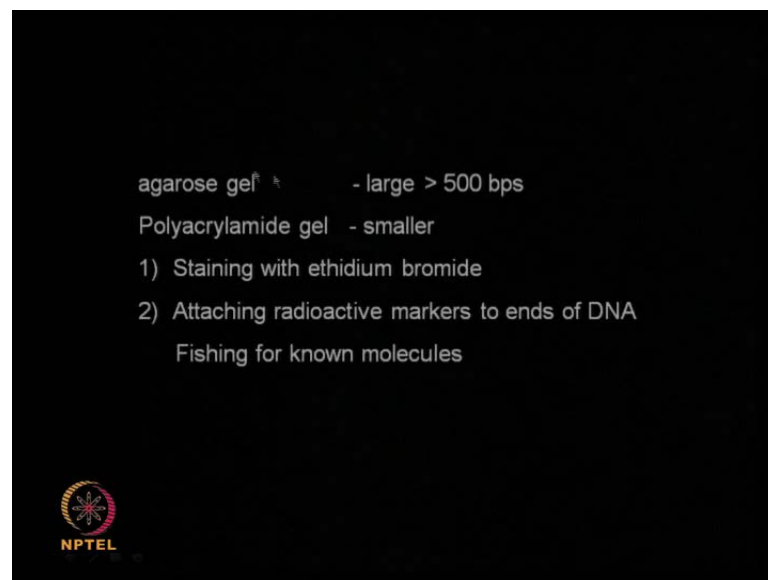


Another operation which is performed on DNA molecules is to separate them by length. For that what is known as gel electrophoresis is used, here in a table or an a tray gel is used and with a comb a little small pits are made. And then the DNA is poured into this, the solution containing the DNA strands is poured into that and then a current is applied,

here you have the negative and here you have the positive electrode. And after some time the strands they try to move in this direction and when they move in this direction, you find that the longer strands move slowly whereas, the smaller strands will move very fast. So, small strands will move fast and long strands will be moving slowly.

So, by calibrating it in a proper manner you can select DNA strands, which are of a particular length suppose, I have a strands of length 80 120 and 160 say. Then I want to choose only strands of length 120 I apply the current in this gel tray, then this will be shortest strand 80, this will be 120, this is 160. So, in this portion I select and take the corresponding DNA molecules. So, this is called gel electrophoresis.


(Refer Slide Time: 20:58)



agarose gel - large > 500 bps
Polyacrylamide gel - smaller

- 1) Staining with ethidium bromide
- 2) Attaching radioactive markers to ends of DNA

Fishing for known molecules




NPTEL

(Refer Slide Time: 21:19)

adenine A DNA (deoxyribonucleic acid)
guanine G
cytosine C
thymine T

3' : AAGCTCAG . . . 5'
5' : TTCGAGTC . . . 3'
 $\Sigma = \{A, G, C, T\}$

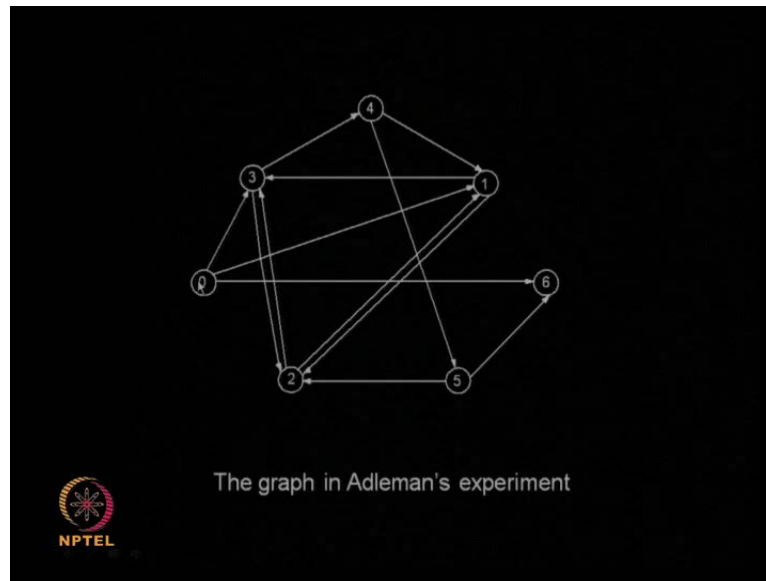
The practical possibilities of encoding information in a DNA sequence and of performing simple bio-operations were used by Adleman to solve a 7 node instance of the Directed Hamiltonian path problem. A directed graph G with designated vertices v_{in} and v_{out} is said to have a Hamiltonian path if and only if there exists a sequence of compatible one way edges e_1, e_2, \dots, e_n (that is, a path) that begins at v_{in} and ends at v_{out} and enters every other vertex exactly once.



So, smaller one you use Polyacrylamide gel and for the larger ones you use Agarose gel. And how do you note it? You can stain with some fluorescent object and then deduct where they are. Anyway those are details of biochemistry and we will not go into that, we will more look into the algorithmic part of it. Now, Adelman performed an experiment first in 1994 and that was the starting point for this subject or this new model of computing. And what he took was a graph, directed graph and the problem is to find out whether that directed graph had a Hamiltonian path.

So, the practical possibilities of encoding information in a DNA sequence and performing simple bio-operations were used by Adelman to solve a 7 node instance of directed Hamiltonian path problem. What is a Hamiltonian path? A directed graph G with designated vertices v_{in} and v_{out} , v_{in} is the initial vertex v_{out} is the final vertex. It is said to have a Hamiltonian path if and only if, there exists a sequence of directed compatible one way edges e_1, e_2, e_3 etcetera, that begins at v_{in} and ends at v_{out} and enters every other vertex exactly once.

(Refer Slide Time: 22:40)




So, this is the graph he took, there are 7 nodes 0 1 2 3 4 5 6 etcetera. There is a directed edge between 0 and 6, there is a directed edge between 4 and 5 and 3 and 2 1 and 3 and so on. The input vertex is v_{in} is 0, the final vertex is 6 that is v_{out} is 6 does there exist a directed path from 0 to 6 which passes through all the other vertices exactly once. You can very easily see that in this there exists a path 0 to 1, 1 to 2, 2 to 3, 3 to 4, 4 to 5 and 5 to 6. In some graphs you may have more than one path, like that in this particular graph there is only one path one Hamiltonian path. The question whether a graph has a Hamiltonian path is an NP complete problem.

So, it does not seem to have a deterministic polynomial time algorithm, but using DNA strands, you may solve it in linear time and that is what we want to see. In a new manner to find out whether, there is an Hamiltonian path from 0 to 6, you arrange the vertices 0 to 6 in the order, first one should be 0 and 6 and in between the 5 vertices you can have any permutation. So, there will be 5 factorial permutations and each one of these permutations you can check whether, it constitutes a directed path from 0 to 6. This is an exhaustive search and it cannot be done in polynomial time.

(Refer Slide Time: 24:48)

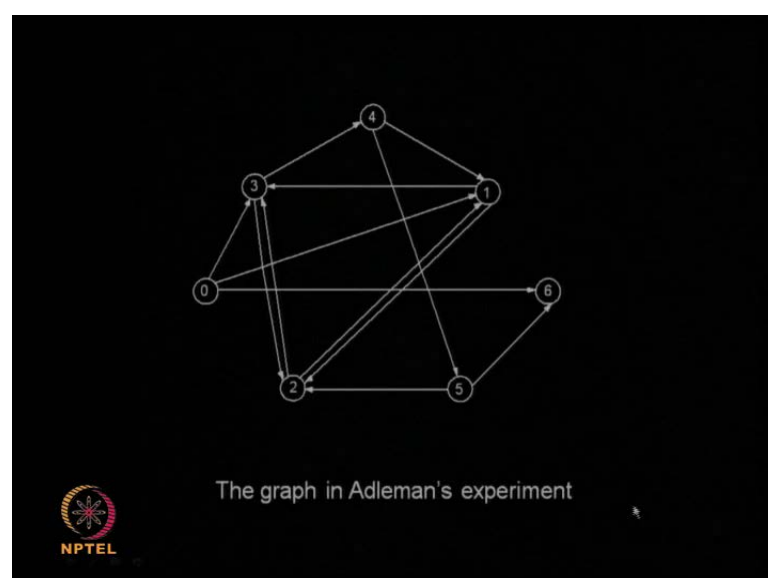
The following algorithm solves the problem

- Step 1. Generate random paths through the graph
- Step 2. Keep only those paths that begin with v_{in} and end with v_{end} .
- Step 3. If the graph has n vertices, then keep only those path that enter exactly n vertices.
- Step 4. Keep only those paths that enter all of the vertices of the graph at least once.
- Step 5. If any paths remain say "YES"; otherwise say "NO".



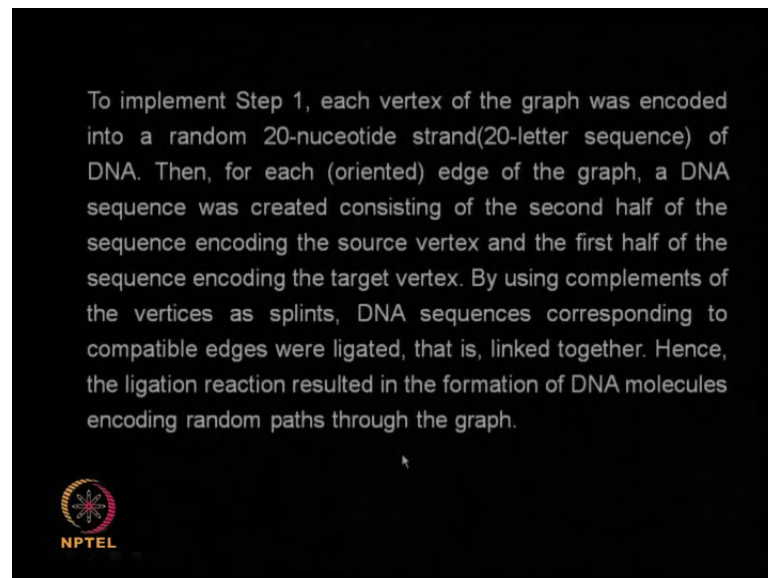
Now, the algorithm he designed for this problem was like this, this is not the usual type of an algorithm, it had five steps. Step 1 is you generate all random paths through the graph and then second step will be keep only those paths that begin with v in and end with v end. Third portion will be, if the graph has n vertices, then keep only those path that enter exactly n vertices including the initial and the final one. Then the forth step is keep only those paths that enter all the vertices of the graph at least once. Then the fifth step is if after all this any paths remain. Then you say yes, there is a Hamiltonian path otherwise you say no.

(Refer Slide Time: 25:45)



What we mean by a random path is like this. See for example, 0 3 2 3 2 3 2 3 4 1 3 such (()) that is a random path, 0 1 then 2 1 2 3 4 5 2 that is another path, like that you may have several paths. So, first of all we have to generate all such paths and in those paths some path may be like this from 4 to 5 then 2, then 3, then 2, then 1, you may have a path like that, but we want to find out the paths between 0 and 6. That is the paths should start at 0 and end at 6, that is the second step of the algorithm keeps only those paths that begin with v in and end with v end. And the third step is, if you have n vertices the length of the paths should be n . So, 0 1 2 3 4 5 6 it the length is 7, then you must make sure that it passes through each one of the vertices of the graph and then finally, check whether, there is a path which satisfies all these criteria.


(Refer Slide Time: 27:15)



Now, how it was done using DNA strands is like this. Now, each vertex of the graph was encoded into a random 20 nucleotide strand of DNA. Then for each edge of the graph a DNA sequence was created consisting of the second half of the sequence, encoding the source vertex and the first half of the sequence encoding the target vertex. By using compliments of the vertices as splints, DNA sequences corresponding to compatible edges were ligated, that is linked together. Hence, ligation reaction resulted in the formation of DNA molecules encoding random paths through the graph.

(Refer Slide Time: 28:01)

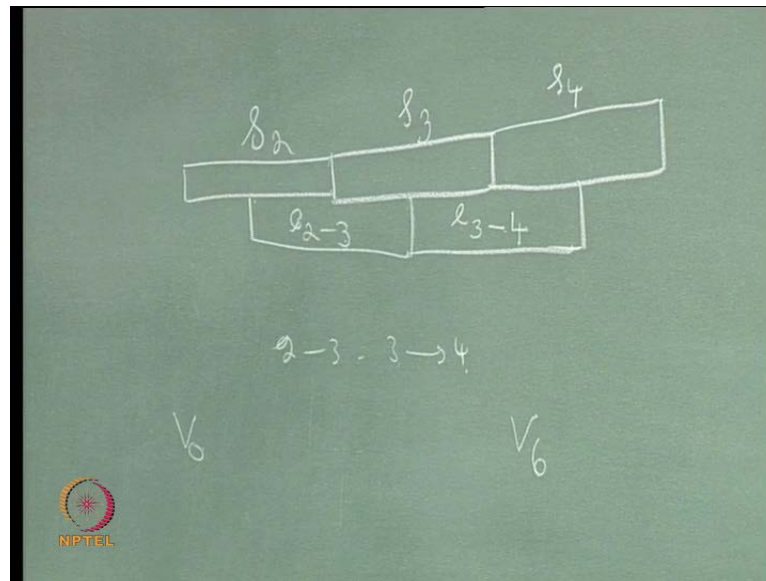
L.M Adleman: Molecular Computation of solutions to combinatorial problems

$$\begin{aligned}S_2 &= \text{TATCGGATCGGTATATCCGA,} \\S_3 &= \text{GCTATTCGAGCTTAAAGCTA,} \\S_4 &= \text{GGCTAGGTACCAGCATGCTT.}\end{aligned}$$
$$\begin{aligned}e_{2 \rightarrow 3} &= \text{CATATAGGCTCGATAAGCTC,} \\e_{3 \rightarrow 2} &= \text{GAATTCGATATAGCCTAGC,} \\e_{3 \rightarrow 4} &= \text{GAATTCGATCCGATCCATG.}\end{aligned}$$


Let me see, see he took S_2 like this, that is the sequence of 20 letter sequence, representing the vertex 2, another 20 letter sequence representing the vertex 3, another one representing the vertex 4. Then there is an edge between 2 and 3 how is this edge represented? It is from 2 to 3. So, this second half of S_2 is taken this portion and the complimentary sequence is taken, for G is the complimentary C, for T it is A, for A it is T, for T it is A and so, you see that the first portion of the edge is the compliment of this portion. Whereas, the second portion is the compliment of the first portion of this GCTA compliment should be CGAT. So, you see that this portion CGATAA this represented the compliment of the first portion of x_3 .

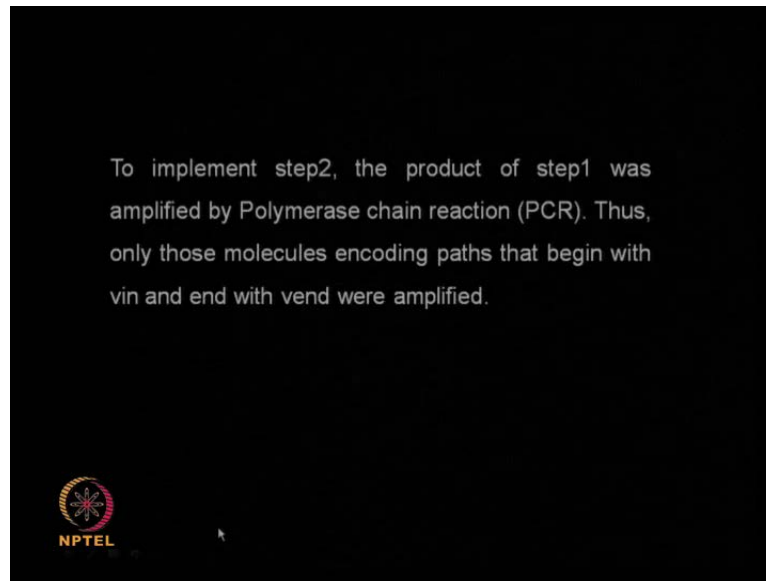
So, similarly, there is an edge between 3 and 2, starting from 3 to node 2 and how is this edge represented, it is represented by a 20 letter sequence or 20 nucleotides. The first 10 will be the compliment of the second part of S_3 , this is the sequence for S_3 , the second half of that is this, the compliment GAA etcetera will be the first part of this. And the compliment of the first part of S_2 , which is a TA, this portion that will be the second portion of the vertex from 3 to 2. Similarly, the edge from 3 to 4 can be defined by a 20 letter sequence. So, what he did was all these DNA strands which are of length 20 which are of length 20 they were taken and put in a test tube and ligase added so, that they could join together.

(Refer Slide Time: 30:39)



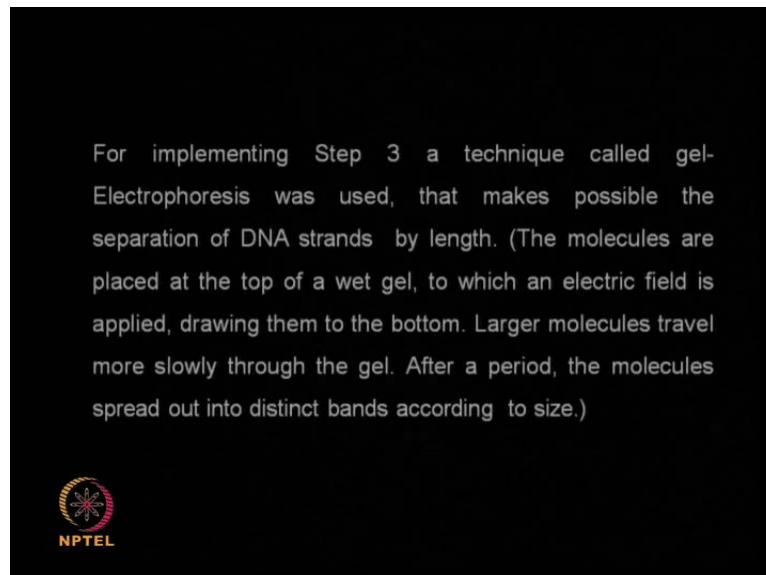
So, suppose there is an edge, there is a vertex S_2 and this is single strand of letter, 20 letter sequences and S_3 is vertex 3, there is an edge between 2 and 3 that will join here edge 2 to 3. And suppose, there is another strand S_4 , the edge corresponding to e 3 to 4 will come and attach like this, because the first portion of it is the complement of the second portion of S_3 and so on. So, strands like this will represent paths what does this represent? It represents an edge from 2 to 3 and then 3 to 4. So, by mixing all the strands together, all the strands which represent the nodes and also which represent the edges you form double strands of this form, which will represent all sort of random paths through the graph. Now, among that, we want to select those which begin with v_0 and end with v_6 node 6 how do we achieve this? Adelman used the following criterion to do that.

(Refer Slide Time: 32:18)



The second step he implemented in the following manner, the product of step 1 was amplified by polymerase chain reaction and there strands using proper primers, the primers will determine where they begin. So, by using v in and v end as primers it was amplified.

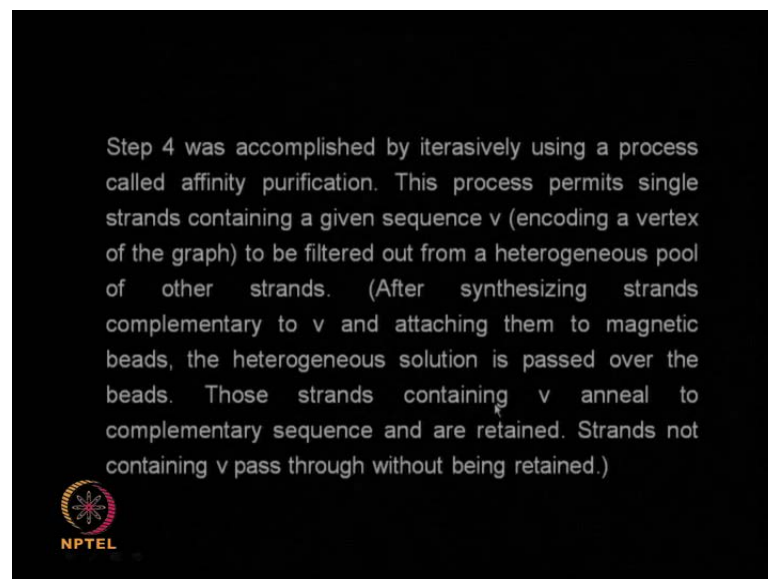
(Refer Slide Time: 32:49)



So, finally, you ended up with only paths which begin with v 0 and end with 6, that is node 0 and end with node 6. How is the step 3 implemented? Step 3 is you have to make sure that all the paths are of length 7 or because the node is represented by a 20 letter

sequence, here the DNA strand should be of length 140, 7 nodes should be represented by 7 into 20 140 length 20 length 140. So, for the implementing step three a technique called gel electrophoresis that we have seen what it is. And you put all random paths, all DNA strands, representing random paths and use gel electrophoresis and select only paths of length 140 that is the length of the path is 7 in terms of the nodes. So, you apply an electric field and larger molecules travel slowly after a period the molecules spread out into distinct bands and you choose only those of size 140.

(Refer Slide Time: 34:07)



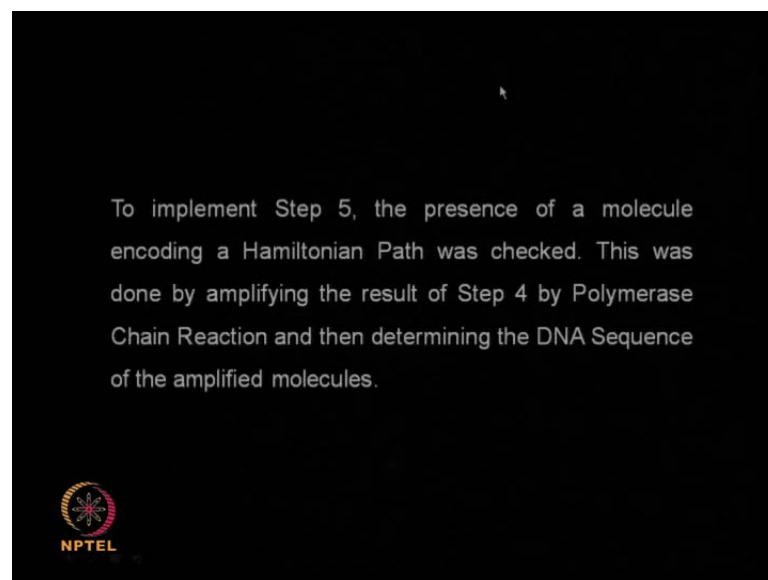
The next step is you have to make sure that the Hamiltonian path passes through every vertex of the graph. So, you have seen that it begins with 0 and it ends with 6, but how do you find out that it passes through 1. For that you use what is known as affinity purification? The compliment of vertex 1 is taken and attached to a glass bead; the whole solution is passed through that. And wherever the sequence corresponding to the compliment of node 1, I mean the node 1 sequence attaches with the compliment, attached to the glass bead. And so, the rest of them will be washed away and these strands you collect after a little bit of heating.

So, this process affinity purification is the process, permits single strands containing a given sequence v to be filtered out from heterogeneous pool of other strands. After synthesizing strands corresponding to complimentary to v and attaching them to magnetic beads, the heterogeneous solution is passed over the beads. Those strands

containing v anneal to complementary sequence and are retained; strands not containing v pass through without being retained. So, after doing this slightly heat the solution so, these single strands will come out, then you make sure that the same thing is applied for vertex 2.

So, that only strands containing the sequence corresponding to vertex 2 remain, then you make sure for vertex 3, then vertex 4 and then vertex 5 that way you make sure that, all strands are of length 7 I mean the terms of nodes it is 7. In terms of DNA strand it is 140, then it passes through v_1 , it passes through v_2 , it passes through v_3 , it passes through v_4 , it passes through v_5 . That way if it starts in 0 and ends in 6 and passes through 1 2 3 4 5, then it must be a Hamiltonian path.

(Refer Slide Time: 36:40)



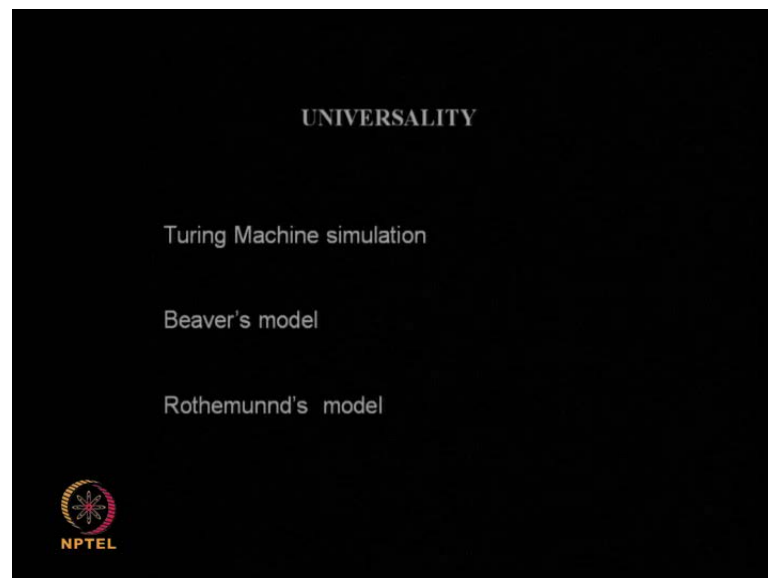
Now, do we have a DNA strand which satisfies all these conditions, how do you test it? In the last step if at all the solutions contain some paths some DNA strands representing paths from 0 to 6, then that must give you the answer yes, but then in performing these operations the number of DNA strands may be reduced. So, to make sure that you deduct all the paths properly, we again apply the polymerase chain reaction to multiply to amplify. If you have 2 or 3 strands by doing this you may end up with 30 40 strands and so, there should not be any problem in deducting that.

So, if at the end you have some DNA strand remaining that will represent the Hamiltonian path. This way Adelman showed that a Hamiltonian path problem, which is

an np-complete problem can be solved using DNA strands. The main point to be noted here is that, all the random paths are parallel generated, there is immense parallelism and that is what is helpful here? Also you can see that the Watson-crick complementarily that is A joins with T and C joins with G and that helps in doing all this experiments and designing the algorithms.

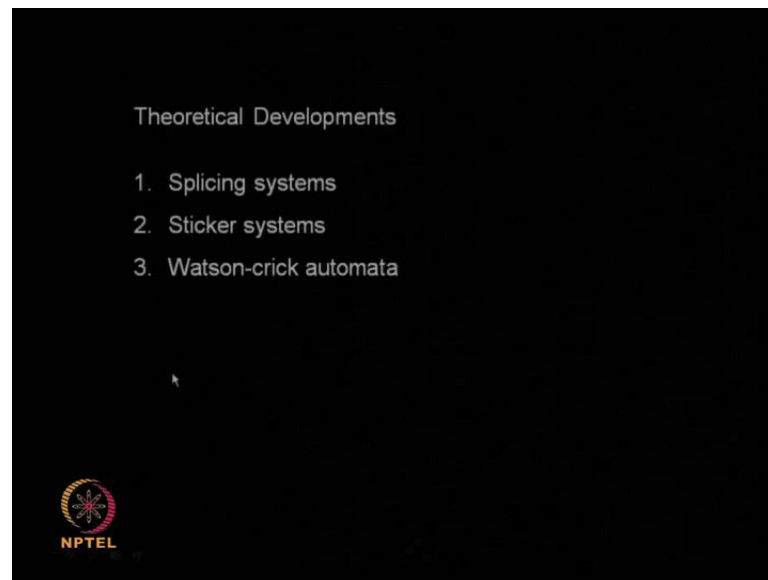
After Adelman designed this algorithm for Hamiltonian path, Lipton came out with an algorithm which is somewhat similar to this, for the Boolean satisfiability problem. You know that Boolean satisfiability is the first problem to be shown to be np-complete. So, for that Lipton came out with an algorithm, we shall not go into that. Afterwards several problems they found that they can be solved with proper DNA algorithms.

(Refer Slide Time: 39:02)



The DNA algorithms are basically different from the ordinary algorithm. So, initially there was a lot of enthusiasm that a biochip can replace a silicon chip and everything can be done by a bio-computer. And at least to prove it theoretically everything can be done by a biochip or a DNA computer, you have to show that the Turing Machine can be simulated with that this has been done. It has been shown that the operations of a Turing machine can be simulated by a DNA computer, there were two models one is called the Beaver's model and another is called a Rothemunnd's model. This model seems to be a slightly better model because it does not involve step by step simulation, but the whole thing is simulated in the in a full.

(Refer Slide Time: 39:58)



Now, this is about the DNA computing. Initially there was a lot of enthusiasm that the bio-computer or a biochip will replace, but afterwards the enthusiasms slightly came down because it is not going to help if for multiplying two numbers. It is going to take 50,000 rupees and 3 days so, you are not going to do that. So, the cost involved and the time involved are not so, helpful in this, but in future something may happen, developments may take place in biochemistry and lot of improvements can take place. So, the things can be done in a much quicker and less costly manner and sometimes for at least for some problems, the DNA computer may prove to be very useful.

Whatever it is in formal language theory, lot of theoretical developments have taken place because of this. New models have been defined for generating languages one of them is the splicing system, we shall briefly see what is a splicing system? Actually this model was defined in 1987 itself, nearly 7 years before Adelman published his work. Another model is sticker systems and in the automata site you have the Watson-crick automata, which deal with double stranded strings.


(Refer Slide Time: 41:51)

What is Recombination of DNAs

Double Stranded DNAs

CGCGCGACGCGCAGC
GCGCGCTGCGCGTCG

ATATAGACGCAATT
TATATCTGCGTTAAA



We shall just have a brief look at what is a splicing system? The recombination of DNA's or recombinant behaviour of the DNA's is used here; you have double stranded DNA's like this. This is one double stranded DNA, another double stranded DNA, you may have a restriction site here, an enzyme may find a restriction site and then cut it at the proper place.

(Refer Slide Time: 42:21)

RESTRICTION ENZYMES CUT THE DNAs AT THE SPECIFIC SITES IN A SPECIFIC PATTERN

CGCGCGACGCGCAGC
GCGCGCTGCGCGTCG

Restriction Enzymes

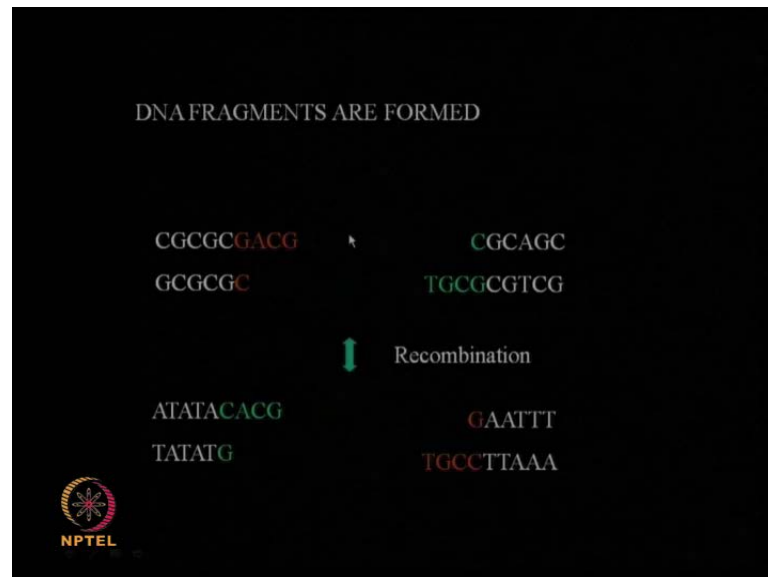
ATATACACGGAATT
TATATGTGCCTTAAA



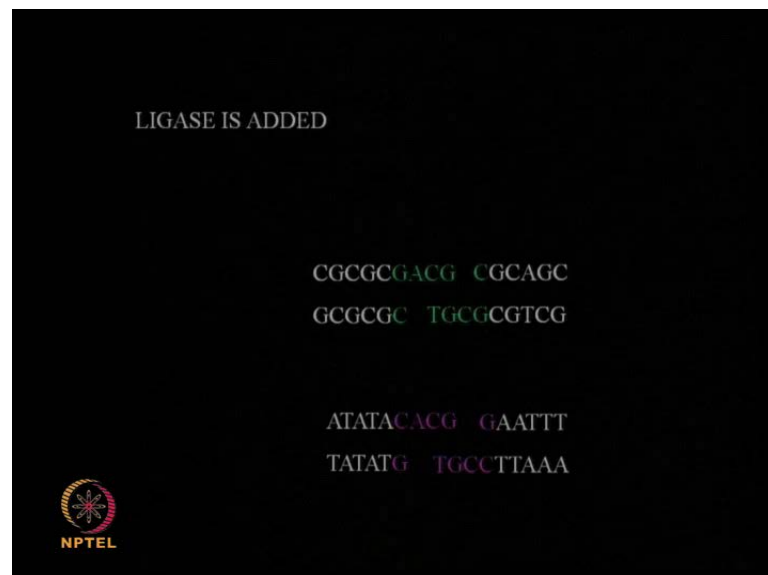
For example, take these two sequences you have a restriction site here, you have a restriction site here and then, the enzyme when you apply it may cut it in a proper

manner like this. So, the first strand is cut into two pieces like this and the second strand is cut like this. Now, you find that you have ACG here; you have the same ACG here and so on. So, this portion can join with this and this portion can join with this and Ligase is added.

(Refer Slide Time: 42:36)



(Refer Slide Time: 43:01)




(Refer Slide Time: 43:12)

TWO NEW DNAs ARE FORMED

ATATACACGCGCAGC
TATATGTGCGCGTCG

Ligated !


CGCGCGACGGAATT
GCGCGCTGCCTAAA

The process of cutting the DNAs using the restriction enzymes
and pasting the DNA fragments using the ligase enzymes is called
 A recombination

(Refer Slide Time: 43:26)

ABSTRACT MODEL OF DNA RECOMBINATION

- Instead of DNAs, strings are taken.
- Two strings (analogous to DNA strands) $W_1 = w_1 u_1 x v_1 w_1'$
 $W_2 = w_2 u_2 x v_2 w_2'$
- Two triplets (analogous to restriction sites) (u_1, x_1, v_1)
and (u_2, x_2, v_2)




So, DNA fragments are formed like this, then when you add the Ligase this joins like this and this joins like this. So, in the end, you get two other strands, this operation is called splicing and formally it is represented as a splicing system. The abstract model is defined like this, instead of using the DNA strand you use any string, any string over an alphabet. And two strings analogous to DNA strands, you have one string w_1 as w_1 dash u_1 x v_1 w_1' w_2 w_1 2 dash, another string w_2 which is w_2 dash u_2 x v_2 w_2' w_2 dash. Then you see that in the middle you have u_1 x v_1 and you have u_2 x v_2 , these triples have taken as restriction site, these can be looked it as restriction sites here. This is the

way Tom head defined and because of his name such systems are called h-systems h-scheme etcetera. (No Audio From: 43:28 to 44:34)

(Refer Slide Time: 44:33)

FORMAL DEFINITION OF SPLICING OPERATION

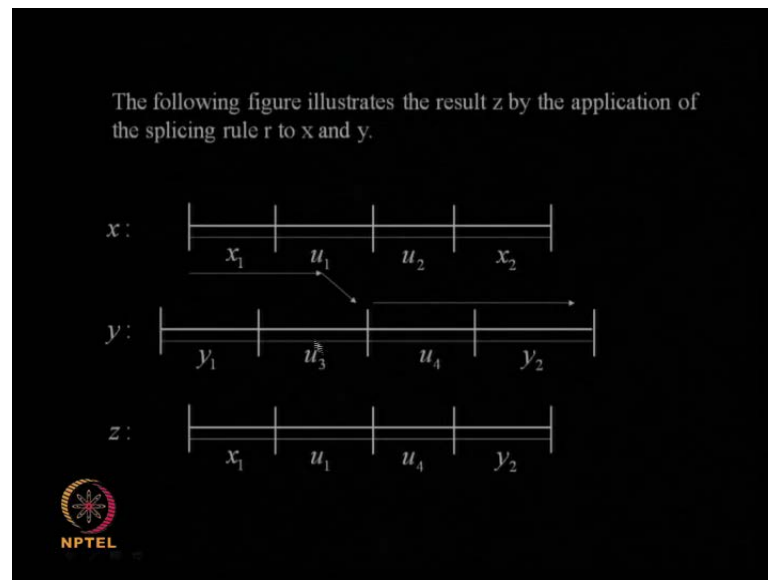
Consider an alphabet V and two special symbols, $\#$ and $\$$. A splicing rule r (over V) is a string of the form $u_1 \# u_2 \$ u_3 \# u_4$, where $u_1, u_2, u_3, u_4 \in V^*$. Given such a rule r , and $x, y, z \in V^*$, we write $(x, y) \vdash_r z$ iff

$$x = x_1 u_1 u_2 x_2, \quad y = y_1 u_3 u_4 y_2, \quad z = x_1 u_1 u_4 y_2 \quad \text{for some } x_1, x_2, y_1, y_2 \in V^*$$


The latter definition they do not make use of that x . Whereas, generally an alphabet v is considered and the rules splicing rules are represented like this, $u_1 \# u_2 \$ u_3 \# u_4$. Where hash and dollar are two special symbols and $u_1 u_2 u_3 u_4$ are symbols or strings over the letter v , this is called the splicing rule you call it as rule r . So, it is of the form $u_1 \# u_2 \$ u_3 \# u_4$, some of them can be empty strings also. So, they are strings over V star.

And given such a rule, you have three strings $x y z$. From x and y by applying the rule r you get the string z by splicing the following happens. You have x as $x_1 u_1 u_2 x_2$, y as $y_1 u_3 u_4 y_2$ and the rule is this. So, a cut can be made between u_1 and u_2 . So, here you can make a cut here and similarly, you can make a cut between u_3 and u_4 so, you can make a cut here. And then the first portion $x_1 u_1$ can combine with the second portion here to give you $x_1 u_1 u_4 y_2$.

(Refer Slide Time: 46:14)



So, the situation is like this, x is of the form $x_1 u_1 u_2 x_2$ like this, then y is of this form $y_1 u_3 u_4 y_2$ by the rule, rule is this. Please note that the rule is $u_1 \# u_2 \# u_3 u_4$; that means, the cut can be made between u_1 and u_2 , the second string the cut can be made between u_3 and u_4 . So, with this you will find that, the cut is made here, the cut is made here. And the first portion $x_1 u_1$ combines with the second portion $u_4 y_2$, to give the string $x_1 u_1 u_4 y_2$, you could also have this portion combining with this. In the formal definition, these 2 have a slightly different one splicing two splicing you call them.

(Refer Slide Time: 47:12)

SPLICING SCHEME

A pair of $\sigma = (V, R)$ where V is an alphabet and $R \subseteq V^* \# V^* \$ V^* \# V^*$ is a set of splicing rules, is called an H-scheme. Note that, R can be infinite and moreover it is itself a set of strings, hence a language. Then we can consider the complexity of R , i.e., its place in the Chomsky hierarchy. If $R \in FL$ for a given family FL of languages, then we say that H-scheme σ is of the FL type, i.e., if R as a language is regular, Then the H-scheme is called a regular splicing type.

Splicing scheme is known as H-scheme.

For a given H-scheme $\sigma = (V, R)$ and a language $L \subseteq V^*$, we define

$$\sigma(L) = \{z \in V^* \mid (x, y) \vdash_r z, \text{ for some } x, y \in L, r \in R\}$$

$\sigma(L)$ is the language obtained by a one step splicing of any pair of strings from L with respect to any rule from R .

NPTEL

Anyway this is only a brief introduction to the splicing system and we are not going into the details of the definitions. So, what is a splicing scheme? A splicing scheme, it is denoted by a sigma, it has an alphabet and a set of rules. The alphabet you know is it can be anything in the case of DNA it will be double stranded A G C T etcetera. R is the set of rules, which is of the form over the alphabet V plus, V plus etcetera, V star V star it can be lambda also. Hash denotes the cutting place; dollar separates the first string from the second string.

The set of rules it is called the H-scheme why H? H there are two reasons one is this, when you have these two strings and the cut is made and you are joining here, this is like a H turned 90 degrees so, here you join. So, that is a H turned through 90 degrees or it is also because of the name Tom Head who first proposed the formal model. And for a given H-scheme what is the language generated? You look it as a language generated, you start with the set of axioms R set from set and from them take two strings and combine them, you will get another string. So, the set of all strings formed like that gives you the language, this is single iteration in general you can do it several times.

(Refer Slide Time: 49:04)

SPLICING SYSTEM

Splicing system is a triple (V, L, R)


V: A finite alphabet

L: Set of strings in V^*

R: A finite set of splicing rules of the form $V^* \# V^* \$ V^* \# V^*$

Language generated by the splicing system is the set of strings formed by splicing any two strings, using the rules of R

Thus, splicing system is a language generating device .



So, you have iterated splicing, anyway that definitions also I am not going into the detail. That is called H-system and then you can distinguish between total alphabet and target alphabet and call as extended H-systems. So, splicing system is a triple, V is a alphabet, L is the set of strings with which you start and R is a set of rules and they are represented

like this. They can be finite or infinite and we can look them as regular context free etcetera. With this you generate the language.

(Refer Slide Time: 49:35)

ILLUSTRATIONS

Suppose $\sigma = (\{a, b\}, R)$ where $R = \{a\#b\#a\}$

For $L = \{a^n b^n, b^n a^n \mid n \geq 1\}$,

$\sigma(L)$ is computed as follows:


$$(ab, ba) \vdash_r aa$$

$$(a^2 b^2, ba) \vdash_r a^3$$

$$(a^n b^n, ba) \vdash_r a^{n+1}$$

$$(a^n b^n, b^m a^m) \vdash_r a^{n+m}$$

Hence $\sigma(L) = \{a^n \mid n \geq 2\}$ which is regular.



Let us consider one example, the example here is this, you have the alphabet a and b and the rule is a hash b dollar b hash a. So, in the first string, the cut will be made between an a and b, in the second string the cut will be made between b and a. And you start with a language a power n, b power n and b power n, a power n. So, if you take any a b b a you will get a a because the cut will be made here, the cut will be made here and the first portion will combine with the second portion here. Similarly, if you take a squared b squared and b a, the cut will be made here, because the cut has to be made in the first string between a and b, in the second string the cut will be made between b and a.

So, this a squared will join with this a to give you a cubed. Similarly, we can get any string and you find that any string of a's can be obtained, for n greater than or equal to 2. So, the language generated by the system, starting with this language or the axioms with which you started this, with this if you performs single splicing operation. It is not iterated splicing, but it is just only one splicing operation, then you find that you can get this particular languages.

(Refer Slide Time: 51:05)

ILLUSTRATIONS

EXAMPLE

Let $V = \{a, b, c\}$ Let $r = a\#b\#c\#a$. Let $x = a^{10}b$ and $y = ca^5$


Applying r to x and y

$$(aaaaaaaaaab, caaaaa) \vdash_r a^{15}$$

In general for $x = a^m b, y = ca^n$,

$$(a^m b, ca^n) \vdash_r a^{m+n}$$

This rule produces, for a suitable input of two integers m and n (represented in unary) $m+n$ (in unary).



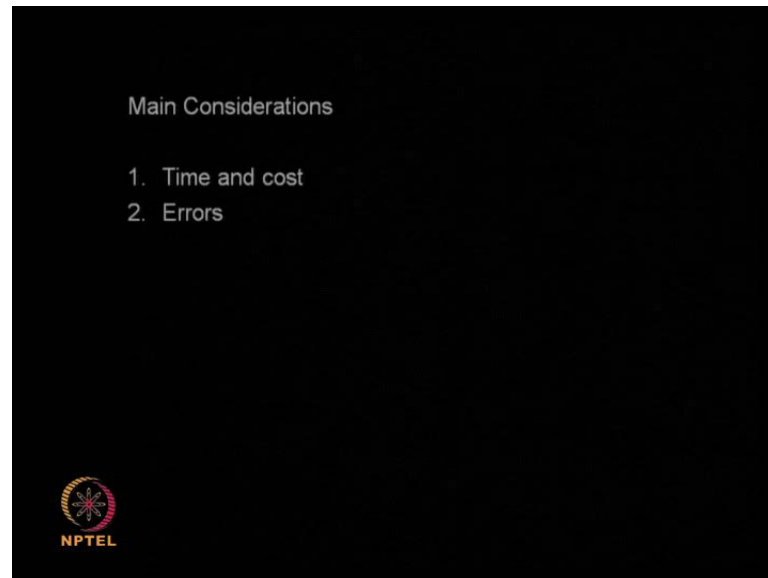
You can also look at it as computing a function, look at this a b c alphabet rule is of this form cut is made between a and b and the second string cut is made between c and a. And you start with x as a power n b and y as c a power n , then if you take a string of this form, the cut will be made between a and b and the second portion the cut will be made between c and a. So, the first portion a power m will combine with the second portion to get you a power m plus n . So, actually it is performing addition, if you have m and n in unary. In unary it is represented as a a a m times, in unary number n is represented as n a's. So, you get the addition in unary it performs addition in unary.

(Refer Slide Time: 52:04)

$$aaaaaaaaaaaaaaaa$$
$$c b$$


For example, $(())$ taking this a power 10 b and c a power 5, you will get a power 15. Obvious this obtained this goes here, this goes here. So, a cut is made between a and b, a cut is made between c and a. And this joints with that to give you a power 15 which is the addition of 10 and 5. The other portion c can combine with b, but you will be do not bother about that.

(Refer Slide Time: 52:41),



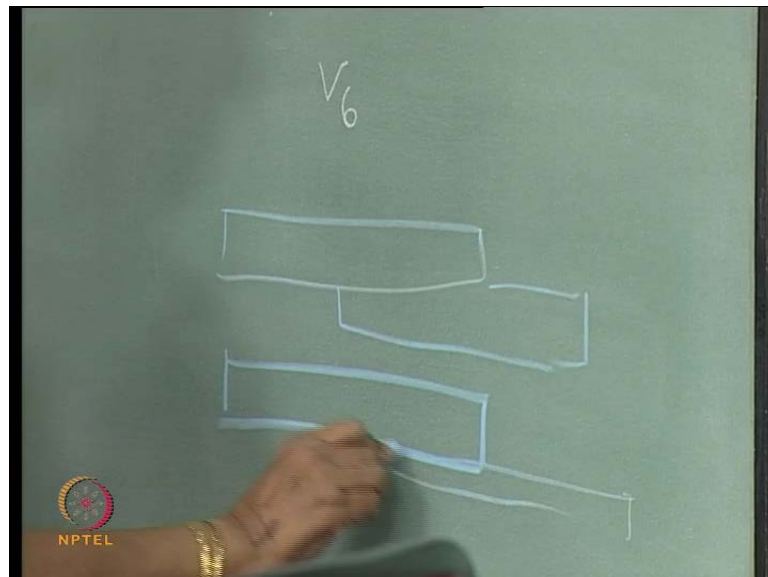
So, can even look at it as computing some function and what are the limitations and main considerations in DNA computing? The main considerations are time and cost, as I told you if it is going to cost you 50,000 rupees and 3 days to the perform, the addition of 2 numbers or multiplication of 2 numbers it is not worthwhile doing it. But this may get improved over the time, because when the first computer was built, it was very big. And it occupied some very big hall, but now you find that, you have very small computers, you have laptops and so on. So, the time and the cost involved in that may become, time may improve and cost may come down, but that may take some time.

The second one is more important there may be lot of errors, because of practical errors involved in doing the experiment. You know that when we considered the Hamiltonian path problem, we assumed that all the random path could be formed. And when we performed the gel electrophoresis proper strands could be chosen and so on. But there be error that you may leave out and so on. What types of error can creep in? Two types of error can creep in. One is you may miss out some strands that is very dangerous.

Sometimes you may not select the proper strands you want. Sometimes you may select unwanted strands. Selecting unwanted strand is not so serious because next step they get discarded, but not selecting wanted strands is serious.

Because you may miss out, that is why from one step to another step when you go in order that you may not miss out the strands. Frequently you perform polymerase chain reaction and amplify. So, when you amplify the number of strands here, get more and more strands you get. And so, the chances of losing out correct strand will be less. Another type of error which can creep is unwanted annealing.

(Refer Slide Time: 55:39)



That is I want something to get attached like this, to one strand I want another string to get attached like this. Whereas, this strand may be like this, this may come and attach somewhere else. That can happen only if you do not choose proper sequences, the sequences should be chosen properly.


(Refer Slide Time: 56:08)

Main Considerations

1. Time and cost
2. Errors

Where it will be useful

1. Associative memory
2. DNA2 DNA computation
3. Cryptanalysis




Suppose, I cannot take a a a a 20 times, because that may create problem. Any subsequence should not have 5 a's or 5 t's and so on. And also something should not get repeated; if you follow certain rules like that these errors could be minimized. And as I told you, it looks that it may not be really useful in every aspect, but on certain aspects it may become very useful. They are associative memory that is lot of information can be stored in one c c of DNA strand. A DNA strand can store a lot of information and within a few c c's of the solution we can show.

(Refer Slide Time: 57:01)

DES 56 bits key
64 bits block
 |
 | encoded
 |
 64 bits

add data

2^{56}



The, if they are taken a single strand one portion may represent the address. It is something like that, address is here and the data is here. So, when I want to fetch the data from here, if I add a complimentary strand for the address I want to fetch the data, which has this particular address. So, the compliment of that can be synthesized and added to this solution and you pick out those strands with this address, then you can find the sequence and find out the data from that. This may take time, but anyway the fact to be noted is it can be done in a proper manner and also within a small space you can store a lot of information. And again with the improvement in biochemistry this may be done in a very short time.

Another one is DNA to DNA computing, in many cases there are tests to be done on DNA. A crime occurs and you want to find out whether the blood there is the same as the blood of the culprit and you suspect somebody to be the culprit and you want to test his blood. So, sequencing is done DNA test is done and so on. In many other cases also in finding out, the crimes and legal battles finding out the DNA sequence becomes important and for that if you use the ordinary computer it may take a lot of time.

Whereas, for such operations DNA to DNA computation. If you want to find out whether a particular subsequence is present in a DNA strand, instead of making them as strings and then testing for the presence of the substring. If you really use DNA computing, this may work out in a better manner and also within a short time you may be able to compute it. So, in these cases DNA computing will be really useful. A third application is cryptanalysis, when you want to break a crypto code, if you take DES, DES has become very old now you have other details. Now, but older one I am mentioning data encryption scheme, you use 56 bits as key this is the old scheme, new schemes you use one 128 bits and so on.

So, 64 bits of block a block has 64 bits, it will be encoded as another 64 bits and there are several strips involved in that using that key. Now, with 56 bits there are 2^{56} possible keys and if you want to find out the key one by one you have to test, anyway I am not going into the details of this. This can be done in a very very efficient manner using DNA computing, because you can test whether a particular bit string is a key simultaneously for all the 2^{56} keys, you can test it simultaneously. So, the time involved will be very less, because of the immense parallelism used in DNA computing.

So, even though, it may not work out well for all problems. Particular problems like this or particular applications like this may find DNA computing as a very useful tool. Time only has to tell us whether such a thing will happen and how far it will become useful? But whatever it is, it has generated lot of interest in formal language and automata theory. By the development of new models like splicing systems, sticker systems and Watson-crick automata. There are other models like Watson-crick L systems and so on. So, this development in DNA computing has enriched formal languages and automata theory.