J. Indian Inst. Sci., July–August 1997, **77**, 283–326.

Computational molecular biology: A survey of problems and tools

LAXMI PARIDA
Courant Institute of Math. Sciences, New York University. email: parida@cs.nyu.edu

1. Solving biological problems using computers

We are interested in computational problems motivated by molecular biology. The problems are interesting and practical solutions are much needed. Most of the problems one runs into, almost without exceptions, are hard. Sometimes, the idealized problems, assuming no experimental error, are not very difficult–consider the problem of sequencing using k-tuple probes*[1]. This has a polynomial time solution but the presence of experimental errors makes the problem difficult. Thus, in most cases the task then is to devise practical, efficient approximate algorithms.

Needless to mention, it is important to understand not just the problems but the processes that give rise to them. Most of the problems arising today are due to the prevalent DNA (or other) technology. It is quite conceivable that a surprising discovery/invention may change the total nature of the problems within half a decade! For example, the gel electrophoresis technology (see Section 2.9.1) gave rise to *partial digest* and *double digest problems* (see Section 2.11.1) which are undoubtedly interesting computational problems. But came along a new technology called *optical mapping*, which completely bypasses the issue of ordering of segments but has brought in a host of other interesting computational problems.

Keeping the volatile nature of this field in mind, the first section reviews briefly some of the aspects of molecular biology and the current technology employed in the laboratories.

The different computational problems are many and varied – sometimes the problems are fuzzy (What is the objective function in the alignment of trees problem?), and the tools employed novel and controversial (Is genome rearrangement a string problem or an energy optimization problem?). As the reader may already suspect, it is hard to place one's fingers on the *right* set of tools. In the second section we discuss, in some details, some of the tools that have been around for a while and are fairly well studied and what we believe tackles problems close to currently posed computational biological problems.

*This is equivalent to finding conditions under which an Eulerian cycle exists in an intersection graph.

2.3. *Structure of DNA/RNA*

Our focus is on the chromosomes, found in the nucleus, which contain the blueprint for the entire organism. In the following sections we study its structure and the mechanism by which the blueprint is interpreted.

The "genetic material" in organisms is genes, which is composed of deoxyribonucleic acid, DNA. DNA is a very large molecule, and is made of small molecules called *neulceotides*. It consist of two complementary chains twisted about each other in the form of a double helix. Each chain is composed of four nucleotides that contain a deoxyribose residue, a phosphate, and a pyramidine or a purine base. The pyramidine bases are thymine (T) and cytosine (C); the purine bases are adenine (A) and guanine (G). The "side" of the double helix consists of deoxyribose residues linked by phosphates. The "rungs" are made of an irregular order of pyramidine and purine bases. The two strands are joined together by hydrogen bonds existing between the pyramidine and the purine bases: Adenine is always paired with thymine (AT) and guanine is always paired with cytosine (GC). See Figure 2.

RNA is very similar to DNA with the following differences:

1. It is single-stranded, *i.e.* only one backbone, with the bases, is present.
2. OH is attached to $C^{2'}$ of the sugar residue, instead of H, in the backbone, as shown in Figure 2.
3. Uracil replaces the pyramidine base thymine.

Orientation of DNA: Note that the structure of the sugar is asymmetric, *i.e.*, its bottom and top ends (where it is attached to the neighbouring phosphates) are not identical. These are designated 5′ and 3′ to distinguish the two ends. Also the backbone pair is oppositely directed. Thus, the ends of the DNA strands are designated 5′ and 3′.

Size of DNA: How large is the macromolecule? Let us look at its size in terms of the base pairs. The human genome contains 23 chromosomes, each of which consists of approximately 3.6×10^9 base pairs. In contrast, the chromosome of *Escherichia Coli* contains only 4×10^6 DNA base pairs. Also, mitochondrial DNA (mt-DNA)* is about 16×10^3 bases in length, and is circular rather than linear.

DNA can be classified in at least two ways:

- By structure.

1. Repetitive DNA (SINES & LINES): The major human SINE (short interspersed repeated sequences) is the *Alu* DNA sequence family, which is repeated between 300,000 and 900,000 times in the human genome. The function of *Alu* is unknown and the reason for their very high frequency in the human genome remains a mystery. LINE (long interspersed repeated sequences) has a consensus sequence of 6400 base pairs. This is repeated between 4000 and 100,000 times. As with the *Alu* sequence the function of these sequences are unknown.

*As the name suggests, this is nongenomic DNA found in the mitochondria. It codes some 13 proteins in humans; mutation in the mt-DNA is responsible for diseases like *Leber's optic atrophy*, which is transmitted solely by mothers – an example of nonmendelian inheritance.

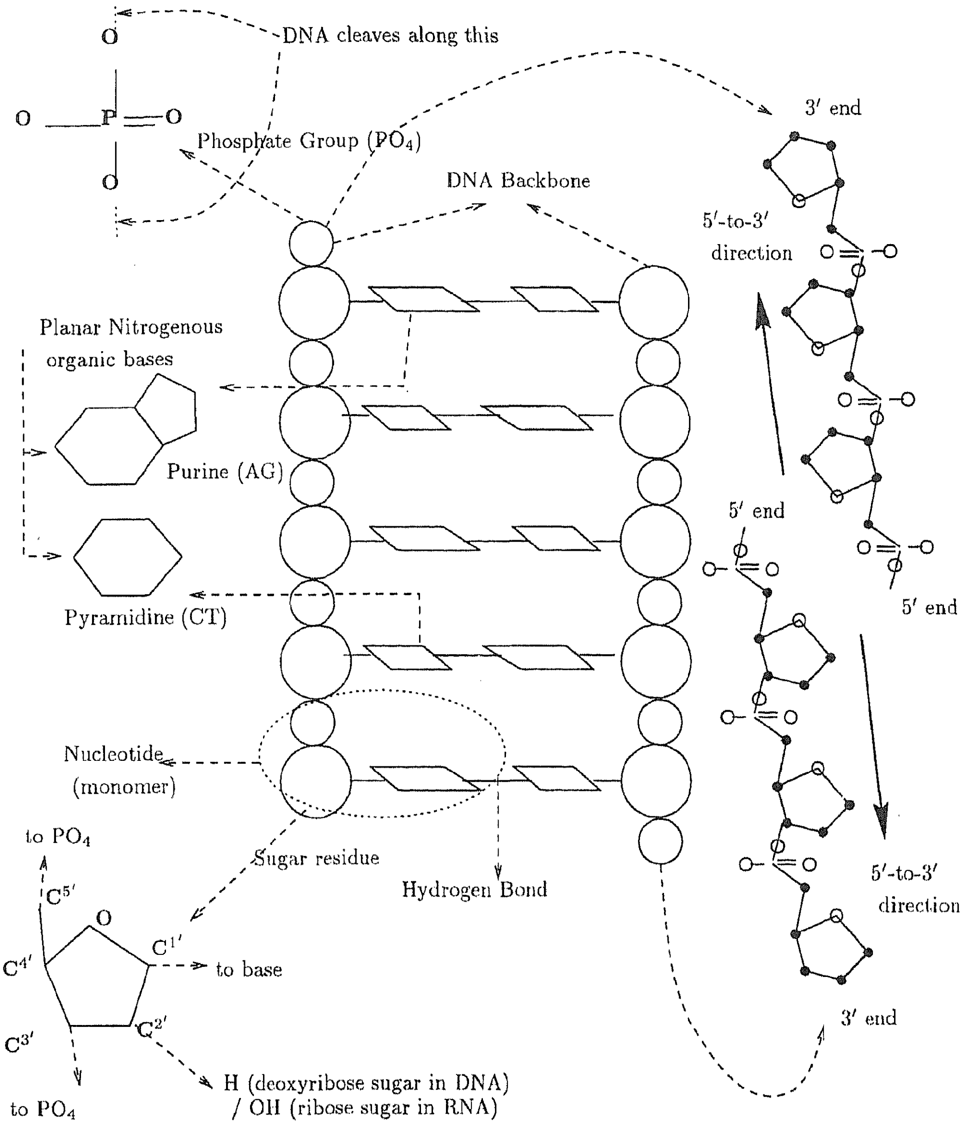

FIG. 2. Structure of DNA. The plane of the planar bases (A,G,C,T) is perpendicular to the helix axis, shown as a string of balls in the picture. Note the opposite directions of the two backbones. The backbone is made of the phosphate group $(PO_4)^{-3}$ and the sugar, $C_5O_4H_{10}$. The base, the phosphate and the sugar form the unit of nucleotide: this is also the unit that is added during the synthesis of DNA.

2. Unique sequence DNA: This contains sequences that code for mRNA. In general, genes are comprised of unique sequence DNA that encodes information for RNA and protein synthesis. mt-DNA consists mostly of unique sequence DNA.

2.3. *Structure of DNA/RNA*

Our focus is on the chromosomes, found in the nucleus, which contain the blueprint for the entire organism. In the following sections we study its structure and the mechanism by which the blueprint is interpreted.

The "genetic material" in organisms is genes, which is composed of deoxyribonucleic acid, DNA. DNA is a very large molecule, and is made of small molecules called *neulceotides*. It consist of two complementary chains twisted about each other in the form of a double helix. Each chain is composed of four nucleotides that contain a deoxyribose residue, a phosphate, and a pyramidine or a purine base. The pyramidine bases are thymine (T) and cytosine (C); the purine bases are adenine (A) and guanine (G). The "side" of the double helix consists of deoxyribose residues linked by phosphates. The "rungs" are made of an irregular order of pyramidine and purine bases. The two strands are joined together by hydrogen bonds existing between the pyramidine and the purine bases: Adenine is always paired with thymine (AT) and guanine is always paired with cytosine (GC). See Figure 2.

RNA is very similar to DNA with the following differences:

1. It is single-stranded, *i.e.* only one backbone, with the bases, is present.
2. OH is attached to $C^{2'}$ of the sugar residue, instead of H, in the backbone, as shown in Figure 2.
3. Uracil replaces the pyramidine base thymine.

Orientation of DNA: Note that the structure of the sugar is asymmetric, *i.e.*, its bottom and top ends (where it is attached to the neighbouring phosphates) are not identical. These are designated 5′ and 3′ to distinguish the two ends. Also the backbone pair is oppositely directed. Thus, the ends of the DNA strands are designated 5′ and 3′.

Size of DNA: How large is the macromolecule? Let us look at its size in terms of the base pairs. The human genome contains 23 chromosomes, each of which consists of approximately 3.6×10^9 base pairs. In contrast, the chromosome of *Escherichia Coli* contains only 4×10^6 DNA base pairs. Also, mitochondrial DNA (mt-DNA)* is about 16×10^3 bases in length, and is circular rather than linear.

DNA can be classified in at least two ways:

- By structure.

1. Repetitive DNA (SINES & LINES): The major human SINE (short interspersed repeated sequences) is the *Alu* DNA sequence family, which is repeated between 300,000 and 900,000 times in the human genome. The function of *Alu* is unknown and the reason for their very high frequency in the human genome remains a mystery. LINE (long interspersed repeated sequences) has a consensus sequence of 6400 base pairs. This is repeated between 4000 and 100,000 times. As with the *Alu* sequence the function of these sequences are unknown.

*As the name suggests, this is nongenomic DNA found in the mitochondria. It codes some 13 proteins in humans; mutation in the mt-DNA is responsible for diseases like *Leber's optic atrophy*, which is transmitted solely by mothers – an example of nonmendelian inheritance.

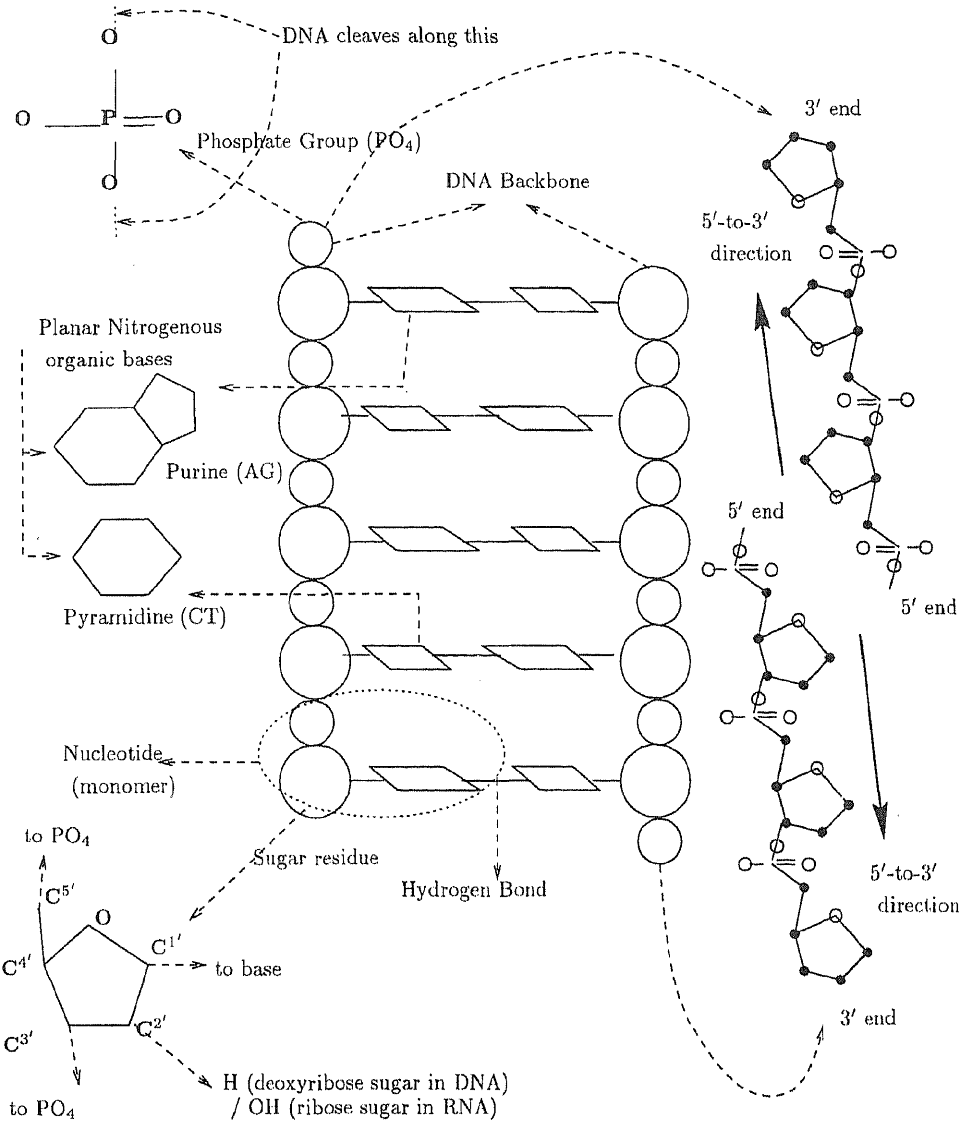

FIG. 2. Structure of DNA. The plane of the planar bases (A,G,C,T) is perpendicular to the helix axis, shown as a string of balls in the picture. Note the opposite directions of the two backbones. The backbone is made of the phosphate group $(PO_4)^{-3}$ and the sugar, $C_5O_4H_{10}$. The base, the phosphate and the sugar form the unit of nucleotide: this is also the unit that is added during the synthesis of DNA.

2. Unique sequence DNA: This contains sequences that code for mRNA. In general, genes are comprised of unique sequence DNA that encodes information for RNA and protein synthesis. mt-DNA consists mostly of unique sequence DNA.

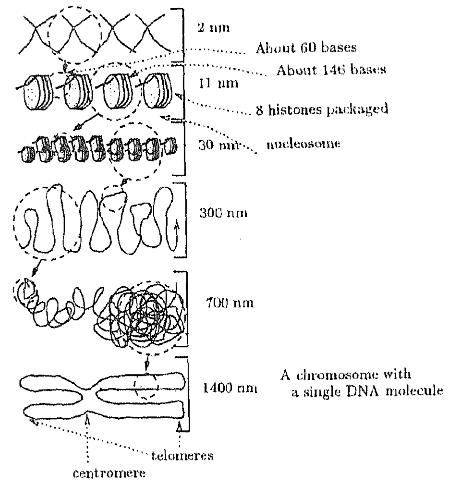

FIG. 3. Packaging of DNA: the figure gives a sense of the scale we are dealing with.

- By function.
  1. Exons: These are the functional portions of the gene sequences that code for proteins.
  2. Introns: These are the noncoding DNA sequences of unknown function that interrupt most mammalian genes.

2.3.1. *Chromosome structure*

The genome of an organism consists of smaller units, *chromosomes*: corn has 20, certain fruitflies have 8 chromosomes, rhinoceroses 84, humans and bats have 46.

Each chromosome contains a single molecule of DNA organized into several orders of packaging to construct a metaphase chromosome: the length of this is about 0.0001 times the length of its DNA. DNA, along with the binding proteins, is called *chromatin*. *Histones* are the structural proteins of the chromatin and are the most abundant proteins in the nucleus. Figure 3 shows some interesting details.

Euchromatin forms the main body of the chromosome and has relatively high density of coding regions or genes. The *chromosome bands* define alternating partitions of euchromatin with differing properties.

- R bands: These stain light with a procedure called the *G banding procedure*. They have a relatively high content of guanine and cytosine, have the majority of SINES (see Section 2.2), and have the highest gene density.
- G bands: These stain dark with the *G banding procedure*. They have a higher content of adenine and thymine, have the majority of LINES (see Section 2.2), and have relatively fewer genes.

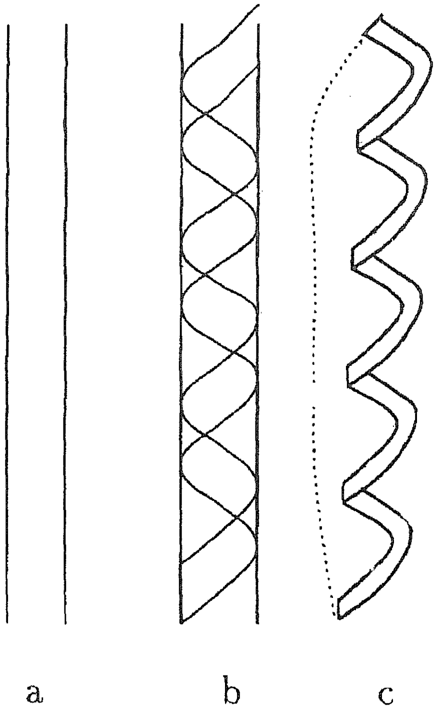

FIG 4. Supercoiling of DNA: (a) A pair of DNA strands with no twists. (b) Helical structure of the pair of DNA strands: *T* is the number of *twists*. (c) In a closed loop, shown by joining the ends of the dotted line, the twisted helical structure can be further coiled (hence called *supercoil*): *W* denotes this twist or *writhe*. The *linking number*, *L*, of a closed DNA loop is defined as $L = T + W$.

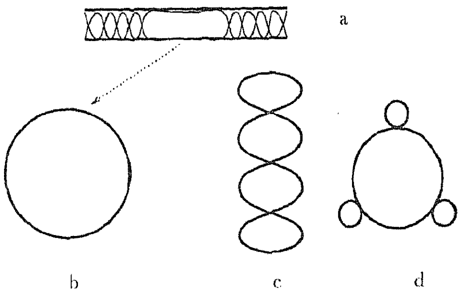

FIG. 5. Conservation of *linking number* of a closed DNA loop: (a) 3 twists of a linear DNA unwound. (b) Loop formed by the DNA shown in (a): its *linking number* has gone down by 3. (c) A possible supercoil structure making up for the loss if 3 in *T*, so that *W* goes up by the same number. (d) An alternate structure to (c).

Heterochromatin is chromatin that is either devoid of genes or has inactive genes.

Each chromosome consists of two parallel strands, the *sister chromatids*, which are held together by a *centromere*. The centromere consists of specific DNA sequences that bind proteins. *Telomeres* are DNA sequences found at the ends of the chromosomes, which are required to maintain chormosome stability. Chromosomes without telomeres that tend to recombime with other chromatin segments are generally subject to breakage, fusion, and eventual loss. The terminal segments of all chromosomes have a similar sequence (TTAGGG), which is present in several thousand copies. Telomere sequences facilitate DNA replication at the end of chromosomes.

DNA supercoiling: Every cell in a complex multicellular organism has identical copies of DNA, yet a liver cell functions differently from a skin cell; say, how does each cell know its different functions? It is quite clear that the base pair sequence information alone does not suffice to explain the functions of the chromosomes. This and many other ques-

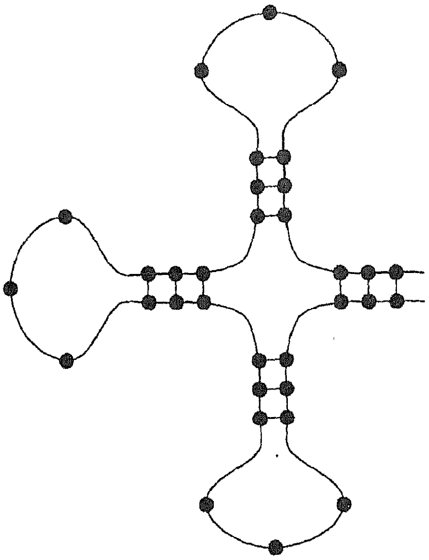

Fig 6. RNA secondary structure: The dots denote the bases, the dotted lines denote the pairing of bases and the solid line denotes the backbone of the RNA. This is a possible RNA secondary structure, termed the *clover leaf.*

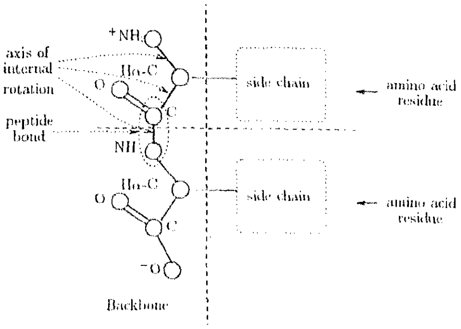

Fig. 7. Linkage of two amino acids forming a single polypeptide. The dotted ellipse shows the peptide bond. The bonds shown in bold are the only three axis of rotation giving rise to a conformation.

tions can, perhaps, be understood by studying the three-dimensional structure of the chromosomes.

It is believed that understanding the highly coiled structure of the DNA strand is critical in defining its three-dimensional structure and its functionality. Figures 4 and 5 explain a simple model based on *linking numbers.*

2.3.2. *RNA secondary structure*

A single-stranded RNA viewed as a linear sequence of nucleotide bases is called the *primary structure. Secondary Structure* is the one obtained by allowing the bases of this single strand to form Watson–Crick pairs without crossing*. Figure 6 shows a secondary structure.

An interesting set of problems is predicting the RNA structure, given the sequence of bases[2].

2.4. *Structure of protein*

Protein consists of amino acids linked together by peptide bonds. The commonly occuring amino acids are of 20 different kinds which all contain the same dipolar ion group $^{+}H_3N{\cdot}C_iH{\cdot}COO-$. The $-NH{\cdot}C_iH{\cdot}CO-$ group derived from it by the elimination of H_2O

*In other words this is a *planar group* which satisfies the following: if a_i pairs with a_j and a_k with a_l, with $i \leq k \leq j$, then $i \leq l \leq j$.

forms the backbone of the polypeptide chain*. See Figure 7. The carbon in the centre is called the α-carbon, α-C. Specificity is provided by the 20 different kinds of side-chains attached to the α-carbon.

Orientation of the polypeptide: As in the backbone of the DNA/RNA, we note that each monomer is really not symmetric: it has –NH– on one end and –CO– on the other, conferring a natural orientation to the chain.

3-*dimensional conformations*: The amino acid sequences of proteins dictate their three-dimensional structures. This is the mechanism by which the one-dimensional genetic code stored in DNA is translated into three dimensions (see section 2.7): Nucleotide sequence enciphers amino acid sequence; amino acid sequence encodes three-dimensional conformation. We cannot predict the conformation of a novel protein structure from its amino acid sequence. However, some general principles of protein architecture have become clear, in that the nature of the interactions that stabilize native conformations, and some of the structural implications have been identified.

2.4.1. *Classification of Protein Topologies*

The most useful classification of families of protein structures is based initially on the work of Levitt and Chothia[19]. Their classification is grounded on the general properties of secondary and tertiary structure in proteins. The hierarchical classification of proteins, or domains within protein structures, is as follows.

1. Primary Structure: A protein molecule may consist of one or more polypeptide chains that, together, may contain between a hundred and several thousand amino acid residues. The longest chain yet discovered is a muscle protein, *titin*, with over 27,000 residues!

   Some proteins contain one or more nonprotein (*prosthetic*) groups which form the sites of their catalytic activity. These may be metal ions or metal-organic compounds.

2. Secondary & Tertiary Structure: Secondary structure refers to the next level of detail of the polypeptide chain; tertiary structure refers to the relationship between these commonly occuring secondary structures.

   At first sight it would appear that a polypeptide chain might be able to assume a very large number of different configurations, but they are subject to the following restrictions:

   - rotation is restricted to a greater or lesser extent about each of the three different bonds making up the chain (see Figure 7), and,
   - no configuration is stable unless it allows every imino group (NH_3) to be hydrogen bonded to a carbonyl (CO) belonging either to the same chain, that is the next amino acid residue, or to a neighboring chain.

   The secondary structures (along with their tertiary structures) are as shown below:

*This is analogous to the backbone of the RNA strand.

(a) α-helical: The polypeptide chain has a helical structure defined by (1) rise per residue, and, (2) angular displacement per residue.

Example: hair and wool.

(b) β-sheet: This is formed by the lateral hydrogen-bonding of different strands. The strands may all be parallel, in the sense of the *orientation* described before, or may be anti-parallel, that is alternate in orientation, or mixed. The sheet may be flat or twisted. Some of the tertiary structure is as follows.

i. parallel β-sheet: Contains two sheets packed face to face in which the strands are almost parallel.
Example: prealbumin.

ii. orthogonal β-sheet: Contains two sheets packed face to face in which the strands are almost perpendicular.
Example: concanavalin.

iii. other β-sheet: The strands of the sheets may be shaped like a "propeller" or a "barrel".
Example: Influenza neuraminidase.

(c) α/β, $\alpha+\beta$: The secondary structure contains both α-helix and β-sheets. The tertiary structure is as follows:

i. α/β: Helixes and sheet assembled from β-α-β units and the strands of sheet are parallel. These are further of two kinds:

A. α/β-linear: The line through centers of strands is roughly linear.
Example: alcohol dehydrogenase.

B. α/β-closed: The line through centers of strands is roughly circular.
Example: triose phosphate isomerase.

ii. $\alpha+\beta$: α-helixes and strands of β-sheet separated in different parts of the molecule (without the β-α-β supersecondary structure).
Example: papain.

(d) Irregular structures: A classification of proteins based on secondary structure must eventually face the structures that contain very few of their residues in helix and sheets. These tend to be stabilized by additional primary chemical bonds, like disulphide bridges in wheat germ agglutinin, or, iron-sulphur clusters in ferridoxin.

(e) Loops: (It is not clear where loops fit in this structural hierarchy.) Loops refer to sections of the polypeptide chain that connect regions of secondary structure. A typical globular protein contains two-thirds of its residues in helix and sheets, and one-third in loops. In many enzymes, loops contain functional residues. Because loops tend to be more flexible in conformational changes than helix and sheets, they are often used when a protein needs to respond to changes in state of ligation.

2.4.2. *Protein Structure Prediction*

When the nucleotide sequences of genes coding for unknown proteins are determined, how is the structure and function obtained? This is not easy to answer. Basically there are two approaches to the problem: the computational approach and the matching approach.

1. The Computational Methods: For the computational model, we may look upon the sequence of amino acids as a string of beads. The input to these computational methods is an ordered sequence of *amino acids* $A_0, A_1, \ldots, A_N$*.

   Let us look at some important properties of the amino acids that are critical in the computational methods. Each bead is either **hydrophobic** (H) or **polar** (P). The H-type monomers (beads) exhibit strong pairwise attraction.

   (a) Energy Minimization Model: This assumes that there exists a potential energy function for the protein and further, that the native or the folded state corresponds to the structure with the lowest potential energy and is thus in a state of thermodynamic equilibrium.

   The three dimensional configuration of the sequence defined by the following:

   - $l_1, l_2, \ldots, l_N$ where l_i is the Euclidean distance between A_{i-1} and A_i.
   - $\theta_1, \theta_2 \ldots, \theta_{N-1}$ where θ_i is the angle between the two segments $A_{i-1}A_i$ and $A_i A_{i+1}$ in the plane defined by the positions of A_{i-1}, A_i and A_{i+1}.
   - $\theta_2, \phi_3 \ldots, \phi_{N-1}$ where ϕ_l is the angle of torsion between the plane defined by the positions of A_{i-2}, A_{i-1}, and A_i and the segment A_iA_{i+1}.

   It is easy to see that the l_i's, θ_i's and ϕ_i's define a unique configuration. The goal is to obtain these $3N-3$ values.

   Let us define the potential energy U, whose global minimum we are seeking as the solution[20].

$$U = L + \Theta + \Phi + C$$

   where,

$$L = \sum_{i=1}^{N} K_l (l_i - l_{0i})^2,$$

$$\Theta = \sum_{i=1}^{N-1} K_\theta (\theta_i - \theta_{0i})^2,$$

$$\Phi = \sum_{i=2}^{N-1} K_{\bar{n}} \left[1 + \cos(\bar{n}\phi_i + \delta)\right],$$

$$C = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{j>i=i+1}^{n} \varepsilon_{i,j} \left(\left(\frac{\sigma_{i,j}}{r_{i,j}} \right)^{12} - 2H_{i,j} \left(\frac{\sigma_{i,j}}{r_{i,j}} \right)^{6} \right).$$

*Recall that there exist about 20 possible amino acids, hence each A_i can have one of the 20 possible values.

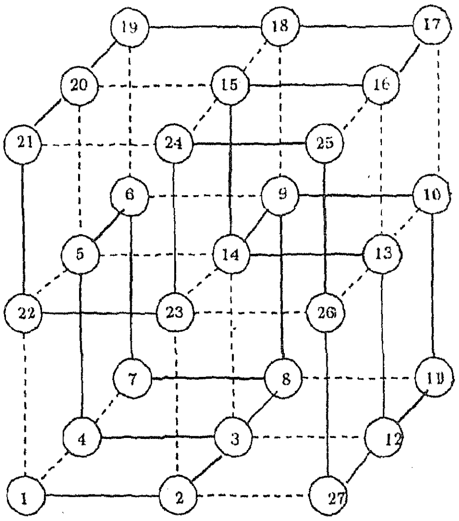

FIG. 8. Lattice model for protein folding. An example of a compact self-avoiding structure of 27 monomers (numbered circles).

K_l and K_θ are the bond stretching and angle bending force constants; l_{0i} and θ_{0i} are the preferred lengths and bends. $K_{\bar{n}}$ is the n-fold torsional constant with a phase shift of δ. This term provides for prefered torsion angles within the molecular structure. If $n = 3$, and $\delta = 0$, the preferred torsion angles are 60°, 180°, and 300°. C is the **Lennard-Jones pairwise potential**. $\sigma_{i,j}$ and $\varepsilon_{i,j}$ are the Lennard-Jones coefficients. $r_{i,j}$ is the Euclidean distance between A_i and A_j. $H_{i,j} = 1$, if A_i and A_j are both H-type and 0 otherwise.

As the reader may observe this function is indeed very difficult to minimize globally. In the literature there are various reports of solving this approximately for simplified models (homo-polymer problems for instance where $A_1 = A_2 = \ldots = A_N$, for small N)^{21}.

(b) *The Lattice Model:*

The number of all possible conformations of a polypeptide chain is too large to be sampled exhaustively, yet protein sequences do fold into unique native states in milliseconds - this is known as the **Levinthal paradox**. Lattice models have been used to address the protein folding problem - one simple model is shown in Figure 8 where the protein chain is a represented as a string of beads on a 3D cubic lattice. The energy of the configuration, U is:

$$U = \sum_{i} \sum_{j,j>i} \delta_{i,j} C_{i,j},$$

where $\delta_{i,j} = 1$ if A_i and A_j are in contact and is 0 otherwise. $C_{i,j}$ is the contact energy as shown in the last section.

The problem can be further simplified by letting a monomer take a position depending on whether it is H-type or P-type and the types of its neighbors. Even this is a hard problem.

2. The Pattern Matching Method: In this the structure is predicted by a process of matching with the known data base. The Protein Data Bank (PDB) is the collection of publicly available structures of proteins, nucleic acids, and other biological macromolecules. Currently, it has around 600 structures determined by X-ray crystallography or neutron diffraction or X-ray fiber diffraction or hypothetical models. The protein of known structure that is closest to the new protein is obtained, by using some distance measures between the amino acid sequences (see section). In the most favorable case, the new protein will be recognizable as a family of proteins of known structure. If, in an optimal alignment, over 25% of the residues are identical, it is fairly safe to conclude that the two proteins have the same fold.

2.5. *Cell Division (Mitosis & Meiosis)*

There are two kinds of cells: *somatic* and *germ*, (also called *gamete*) cells. For lack of a better description, somatic cells are regular cells and germ cells are the reproductive cells. Both the cells, multiply by division, called *mitosis*. Germ cells also have a special cell division called *meiosis*. We shall not get into the details of each of this but give a general overview of the processes.

Chromosomes eluded mankind until early this century, even after the advent of powerful microscopes. It was seen, early this century, that during the cell division process, mitosis, certain structures became visible when appropriately dyed -hence the term *chromosomes*. These condense during mitosis, becoming visible under a microscope.

Meiosis process is more interesting than the mitosis, in the sense that mitosis is an *exact copying* mechanism so far as the chromosomes are concerned whereas meiosis produces some variations, called *cross over*, leading to genetic variations. In this the pair of homologous chromosomes in *diploids**, exchange material from corresponding regions. The cross over information can be used to form a *genetic map* based on traits. Lot of traits are linked, in the sense that these traits are passed on together, as a single package deal, to the offsprings: for example, color of eyes, size of wings and color of the body in *Drosophila*. But occasionally, the traits switch groups and this is attributed to *cross over*. The more often a linked pair get separated, the further they are on the chromosome. Thus a map can be formed for each chromosome, listing the traits (or the genes corresponding to the traits) in linear order with rough distances between them. The unit of this distance was named *morgan***, by the biologist J. B. S. Haldane.

The cell division involves the replication of the chromosomes, that is, the DNA. Let us look at the replication process occuring naturally in living cells and also see how molecular biologists use this knowledge to carry out controlled replication within and without living cells.

*Diploids are those that have two homologous chromosomes.

**Thomas Hunt Morgan and colleagues, early this century, gave a physical basis to the then forgotten Mendelian theory by demonstrating structural relationship between genes and chromosomes.

2.6. *A Simplified Model of Heredity*

The chromosomes can be grouped into pairs: similar members of a pair are said to be *homologous* to one another. As seen in section 2.4.2, there are two kinds of cells: *somatic*, which are *diploids** and, *gametes*, which are haploids**. An *allele*, or gene, is responsible for a trait***, say the color of eye (in *Drosophila*). Most alleles are of two types: *dominant*, say A, and, *recessive*, say a. Each allele type corresponds to an attribute of the trait: for instance, A could be responsible for purple eyes and a for red eyes. Since the allele is present in both the homologous pair, which trait is expressed when one of each is present? The dominant allele, as the name suggests, dominates: trait of a is expressed for the pair (a, a) and trait of A for (A, A), (A, a), (a, A).

After understanding the mechanism of heredity, a natural question that arises is, what are the chances of two (non-twin) siblings, of the same parents, being (genotypically) identical? Assume the organism has N pairs of chromosomes. Let's look at the aspects that bring about genetic variety.

Chromosome combination: A gamete is a haploid, consisting of one chromosome drawn from each of the N homologous pair. Thus there are 2^N possible kinds of gametes for each of the parent. Thus the chance of two siblings having the same genotype is $\frac{1}{2^N \times 2^N} = \frac{1}{4^N}$.

Crossing-over: This is further complicated by the crossing-over between sister chromatids during meiosis. Assuming, on an average, upto k cross-overs occur per chromosome, the number of possible gametes from one parent is d^N where $d = \sum_{i=0}^{i=k} 2^i$ (using a very simplified model). The probability of identical siblings is $\frac{1}{d^2 N}$.

Random Variations: Further, some short sequences of base pairs may undergo changes, that is switch to other bases, due to unexplained reasons.

Significant random variations, or the third factor, bring about *mutation*, and it is believed to be the mechanism of evolution. Thus studying the transformation of the genome of one species (like the mouse, for instance) into another (like the human) has given rise to the (computational) *Genome Rearrangement* Problems[11,12].

2.7. *DNA Replication*

Let us look at the chief actors and the roles they play in the replication process:

- Primer: This is the initiator of the new strand. The usual primer is a very short strand of RNA with four to twelve nucleotides.
- Catalytic Agents (DNA polymerase): An enzyme that catalyzes the polymer formation process.
- A Master template: The parental DNA strand.

*It has pairs of *homologous*, i.e., two copies (not necessarily identical), chromosomes.
**It has only one copy of each of the chromosomes.
***Certain traits cannot be attributed to a single allele pair. Traits that are a result of a single allele pair are called *monongenic*.

- Building Blocks (dNTPs/deoxyribonucleoside triphosphates): As expected, they are of four kinds : dATP, dCTP, dTTP and dGTP corresponding to the four bases.

Starting from a single double-stranded parental DNA molecule, the replication process gives two identical double-stranded daughter molecules. Both the daughter molecules are such that one of the strands is that of the parent and the other is the new synthesized strand.

The replication of the entire strand of DNA occurs in parallel in short strands all over the molecule and merges finally in a rather complex way. Let us look at the steps involved in the replication at a single site, flanked by two *origins of replication*.

1. The parent strand uncoils or *denatures*.
2. The two uncoiled strands replicate.
   (a) A base in the parental strand attaches to the dNTP, by *hydrogen bonds*, containing the complementary base. Thus the dNTP is fixed in position.
   (b) The DNA polymerase catalyzes the creation of an -O-P-O- bridge between the bases, thus forming *covalently bonded* bases.

   Note that the chain grows only in one particular direction, the 5′-to-3′ direction. As a result one strand duplicates continuously but the other (which must proceed in the opposite 3′-to-5′ direction) does so in short strands called *Okazaki fragments*.
3. The two new pairs of strands recoil or *renature* or *anneal*.

How erroneous is the process? The chances are one in a billion that a base in the synthesized daughter DNA would be incorrect!

Protein Synthesis (DNA→RNA→Protein)

This section describes how the "blueprint" is put into effect. Every protein, found in the living body, is synthesized by "executing the program encoded in the DNA".

The protein synthesis follows in the following steps:

1. Transcription: A DNA segment, a *gene*, serves as the template for the synthesis of a single stranded RNA, *messenger* RNA (mRNA). Note that the base Uracil (U) replaces the base T.

   This is similar to DNA replication and requires the essential ingredients: catalytic agent, the RNA polymerase; the Master template, the DNA segment; the building blocks, the NTPs (ribonucleoside triphosphates). Note the absence of the primer.
   (a) The RNA polymerase attaches itself to the *promoter* segment of the double stranded DNA.
   (b) The DNA segment denatures.
   (c) The RNA polymerase facilitates the *hydrogen bonding* of exposed bases with the complementary NTPs. The RNA polymerase further catalyzes the *covalent bonding* between the bases (see DNA replication for more details). Thus the mRNA grows in the 5′-to-3′ direction.

(d) The DNA segment renatures.

2. Splicing: In eukaryotes, both the exons and the introns are transcribed. The resulting primary transcript is spliced; that is, each intron is removed and the exons are linked together.

   This step is absent in prokaryotes. The mRNA leaves the nucleus via the pores and enters the cytoplasm for the next step.

3. Translation: The mRNA serves as a template for stringing together the amino acids in the protein. The succession of *codons* (triplets of adjacent ribonucleotides) determine the amino acid that composes the protein.

   Again, this process is similar to DNA replication and requires the essential ingredients: the ribosome* functions as the catalytic agent; mRNA as the master template. The building blocks are the amino acid monomers, but the process of assembly requires a transfer RNA (tRNA).

   (a) Getting the building blocks ready: A tRNA is a tiny clover-leaf-shaped molecule that has at one end a triplet of ribonucleotides, an *anticodon*, that binds with a complementary codon on the mRNA, and, an attachment site for a single amino acid at the other end. A catalyst, aminoacyl synthetase, converts the tRNA to an aminoacyl-tRNA by attaching the appropriate amino acid to the other end.

   (b) The ribosome travels along the mRNA in 5′-to-3′ direction synthesizing a polymer of amino acids, a protein.

      i. An aminoacyl-tRNA attaches itself to the START codon of the mRNA.

      ii. An appropriate aminoacyl-tRNA attaches to the next codon and the amino acid at its end forms a *peptide bond* with the previous amino acid. The tRNA of the previous one is released. Thus a chain of amino acid is formed with the last monomer still attached to the tRNA.

      iii. The process continues until a STOP codon is reached.

   The ribosome detaches itself and the protein is released into the cytoplasm.

2.9. *Molecular Genetic Techniques*

Let us look at the prevalent techniques for manipulating and analyzing DNA. They can be categorized as:

1. DNA Fragmentation,
2. Fractionating DNA fragments,
3. DNA Amplification,
4. DNA Sequencing, and,
5. Hybridization.

*A very large.molecule composed of ribosomal RNA and at least fifty different proteins.

2.9.1. *DNA Fragmentation (Molecular Scissors)*

Since a single molecule of DNA has about 130 million base pairs, it is important to "chop" the molecule into manageable pieces.

DNA molecules are fragile and mechanical aspects of sample preparation, such as stirring and pipetting, break some of the covalent bonds of the backbones. But the disadvantage is that it is not repeatable, that is, is not expected to break at the same sites. *Restriction Enzymes** are biochemicals capable of cutting the double-stranded DNA, by breaking two -O-P-O- bridges on each backbone of the DNA pair, at specific sequences called the *restriction sites*.

Note that restriction enzyme is a naturally occuring protein in a bacteria that defends the bacteria from invading viruses by cutting up the DNA of the latter. How does the bacterium's own DNA escape the assault? The bacterium produces another enzyme that methylates the restriction sites of its own DNA - this prevents the cleaving action of the restriction enzyme.

Restriction Sites: Let's look at an example to see the effect of the cleaving. The restriction enzyme *Eco*RI recognizes and binds to the palindromic sequence

5′–GAATTC–3′
3′–CTTAAG–5′

If allowed to interact for a sufficiently long time, it cuts the DNA as shown below:

5′–G|AATT C–3′
3′–C TTAA|G–5′

The staggered cuts produce fragments with very "sticky" single stranded ends. These can combine with other matching strands. Some restriction enzymes might cut straight without producing "sticky ends". For example, *Hae*III,

5′–GG|CC–3′
3′–CC|GG–5′

What is the average length of a fragment cut by a restriction enzyme? This is easy to compute: let it be an "n-base cutter", then the pattern occurs on an average every 4^n base pairs. This is the best estimate we have in the absence of any more information: reality might be quite different.

2.9.2. *Fractionating DNA fragments*

The DNA could be fragmented by any method including the one above and fractionated as follows.

1. By Length - Gel Electrophoresis: This is a process whereby the fragments are separated according to their size or electrical charge, on a slab of gelatinous material under the influence of an electric field.

   The phosphate groups in the DNA are negatively charged, hence under the influence of an electric field, the fragments migrate towards the anode. The rate at

*Arber, Smith, and Nathan received the Nobel Prize for their discovery of restriction enzymes in 1978.

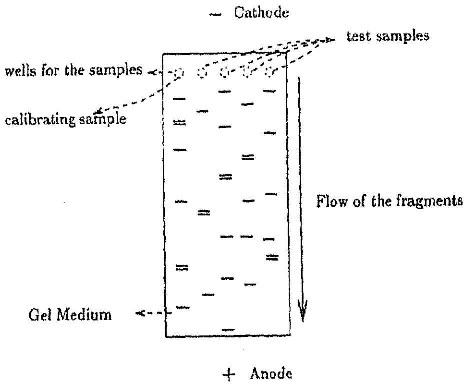

FIG. 9. Gel Electrophoresis: The fragments are separated by lengths. The leftmost known sample calibrates the tracks (vertically) - thus the lengths of the remaining samples can be read off using the first reference.

which it migrates is approximately inversely proportional to the logarithm of its length.

On a slab of gel, wells are made at the top (see Figure 9). The leftmost well contains the calibrating fragments, that is, a sample whose lengths are known. The relative positions of the rest of the columns of fragments with respect to this calibrator give an estimate of the lengths.

Large fragments, over about 50,000 base pairs, do not move well under the influence of steady electric field: *pulsed-field* gel electrophoresis employs a field that is temporarily constant in both direction and magnitude. This solves the problem of large fragments.

2. By Structure - Renaturing: A double strand of DNA *denatures* at around 100°C. When the temperature is lowered the strands randomly *renature*. Rapid renaturing implies high repetitive sequences and slow indicates unique sequence DNA. This technique is used to separate sequences by the repetitive pattern.

2.9.2 *DNA Amplification* (*Molecular Copier*)

Most techniques used in the analysis of DNA rely on the availability of many copies of the segment. Two such methods are:

1. Molecular Cloning: In this method some living cells are used to replicate the DNA sequences. The necessary ingredients are:
   (a) Insert: The DNA segment that is to be amplified.
   (b) Host Organism: This is the the host cell, usually a bacterium, whose replication mechanism is being exploited.
   (c) Vector: This is a DNA segment, with which the *insert* is combined. This is usually a *plasmid*, a nongenomic DNA in the host organism. Some common examples of host organism, vector pairs are shown below.

	Host	*Vector*
1.	*Escherichia Coli* (found in a vertebrate's intestine)	(a) λ phage genome (B) plasmid (natural) (c) cosmid (synthetic)
2.	*Saccharomyces cerevisiae* (baker's yeast)	yeast artificial chromosome (YAC) (synthetic)

The following steps are involved:

(a) Preparing the recombinant DNA: This is done *in vitro*, that is, outside a living cell. The *insert* is combined with the *vector*, say the plasmid. The plasmid is circular, hence it is linearized by digesting with an appropriate restriction enzyme. The DNA strand is digested with the same restriction enzyme, so that the "sticky" ends ligase in the presence of the enzyme, *DNA ligase*.

The rest of the steps are carried out *in vivo*, that is, inside the living cell.

(b) Host Cell Transformation: The *host cell* is exposed to the ligation mixture so that the recombinant DNA may enter the cell. This process is not fully understood, although it can be fairly well controlled.

(c) Cell Multiplication: The solution with the transformed host cells is moved to culture dishes and allowed to multiply in a solid growth medium.

(d) Colony Selection: A colony of cells is produced in the dishes. Note that there is a possibility that the recombinant DNA failed to transform a host cell in step (b). At this step, the different colonies are checked for the presence of the recombinant DNA by various methods (say checking for the expression of a characteristic of the recombinant DNA).

2. Polymerase Chain Reaction (PCR)*: This is an *in vitro* process which is remarkably simple to understand. The requirement is to amplify a segment of the paired DNA. Recall the essential ingredients for DNA replication:primer, catalysts, template and the dNTPs. But here we wish to replicate only certain segment, hence two *primers*, which form the complementary ends of the segment are used. Further, we replicate many times, hence repeated denaturing and renaturing is carried out by changing the temperature. The steps are shown in Figure 10. Every cycle doubles the number of the existing segment, thus the number of cloned segments increase geometrically.

2.9.4. *DNA Sequencing*

We will discuss two such methods, the Maxam-Gilbert Method, and, the Sangar Method** which have the following important characteristics:

*Mullis and Smith received the Nobel prize in 1993 for this technique, which has become a household term after the O. J. Simpson trial.

**Sangar and Gilbert received the Nobel prize, in 1980, for their work on sequencing techniques.

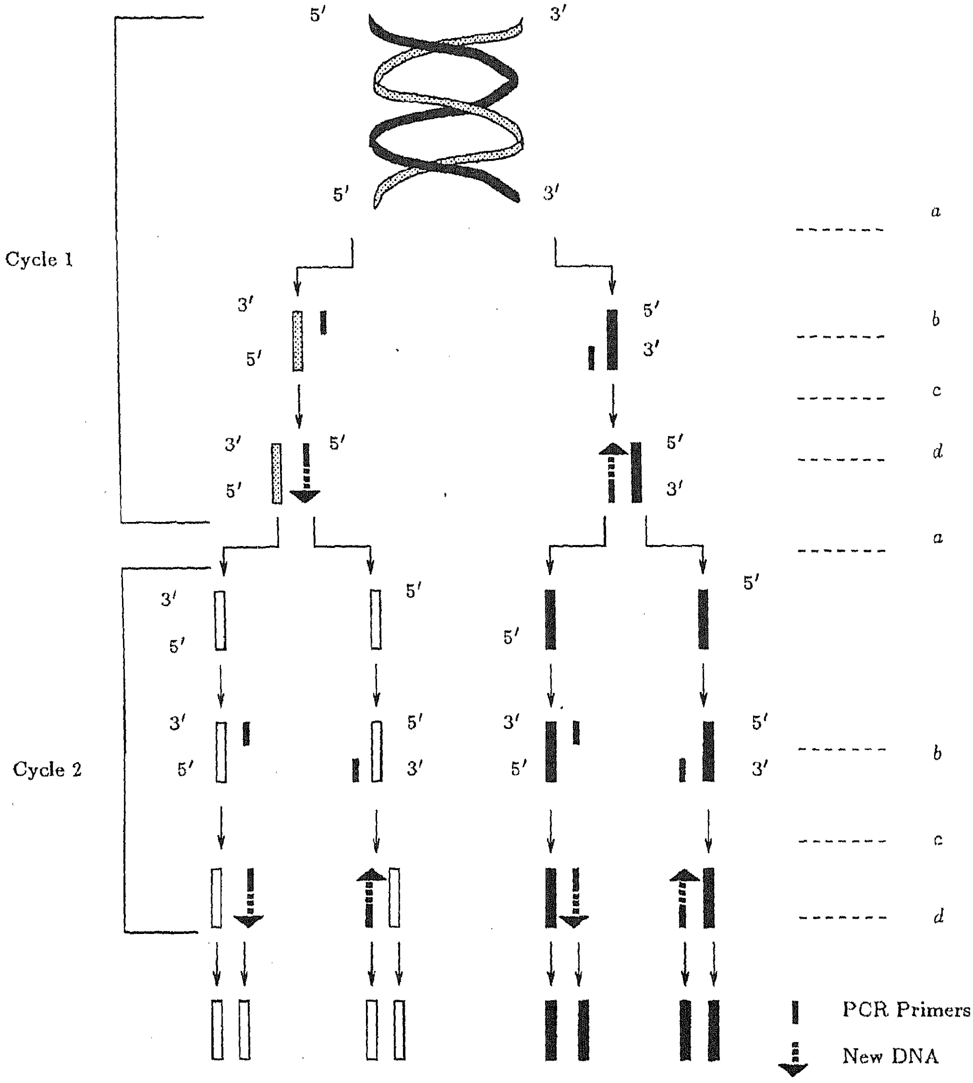

FIG. 10. PCR is based on the amplification of a DNA fragment flanked by two primers that are complimentary to opposite strands of the sequence being investigated. In each cycle, (a) heat denaturation separates the strands, (b) Primers are added in excess and hybridized to complementary fragments. (c) dNTPs and polymerase are added while the temperature increases. (d) The primer is extended in the 3′ direction as new DNA is extended in the 5′ direction.

1. Work on short segments of about 500 to 2000 base pairs.
2. The final step involves reading off the sequence from a radiogram of a gel electrophoresis process, hence it can be done mechanically and thus automatic sequencing machines exist.

For the details of the methods the reader is directed to the references[26]. We briefly sketch the underlying principle: both operate on the ability to identify the base at one end of the segment, with the other end being fixed or labeled. Since the gel electrophoresis fractionates by length, the longest length gives the rightmost base, the shortest gives the leftmost and so on. Thus the rows in the Gel Electrophoresis correspond to the different lengths and the four columns correspond to the four bases A, C, G and T. The natural question is whether Gel Electrophoresis can acutally resolve segments differing in length by a single base: the answer is yes.

- In the Maxam-Gilbert Method, a clever scheme of cleaving at base A or C or G or T is employed. Thus one has four test tubes each with segments cleaved at one of the bases.
- In the Sangar Method, a complementary segment is allowed to grow and stop selectively at the four bases. Thus this newly synthesized strand terminates at the four different bases in a controlled manner in different test tubes.

The samples from the separate test tubes, in both the methods, are used to produce the four different lanes in the gel electrophoretic method.

2.9.5. *Hybridization*

This refers to the hydrogen bonding that occur between any two single-stranded nucleic-acid fragments that are complementary along some portion of their lengths.

If a short fragment of known sequence is labeled with a fluorescent molecule and allowed to interact with denatured chromosomes, the presence or absence of the complementary segment in the chromosome can be ascertained. In particular, in-situ Hybridization (ISH), can be used to locate a sequence in a chromosome, or the position within a single chromosome. Southern Hybridization is used for identifying among a sample of many different DNA fragments, the fragment(s), identified by length, containing the particular sequence.

2.10. *Problems in a Nutshell*

Let us summarize the various computational problems arising in molecular biology in the following table.

	DNA/RNA	*Proteins*
Structure	• The base sequences • The superhelical/secondary structure	• The amino acid sequences. • Configuration 3-dimensional in space.
Function	• Patterns/repetitions etc. in the base sequence. • Locating exons/introns, genes, etc. on the genome.	• Patterns in the amino acid sequences. • Relation of the structure to function.
Applications		Rational drug design.

2.11. *Molecular Computers/ DNA Computing*

We conclude this tour by looking into the possibility of harnessing the DNA molecules for general computing. Let us note two important features of the bio-chemical reactions we have seen so far in this section*.

1. Speed. Chemical reactions occur very fast and in parallel. For example, a single *E. Coli* cell can, under suitable conditions, multiply into more than a billion cells in about 10 hours.

   A typical desktop computer performs about 10^6 operations/second, the fastest supercomputer about 10^{12}; assuming the ligation of two DNA molecules as one operation, about 2×10^{19} operations/second is a very feasible number.
2. Space. The DNA molecules are extremely small! See Figure 3. It allows for information density of 1 bit/cubic-nanometer, whereas conventional storage media like videotapes, store about 1 bit/10^{12} cubic-nanometer.

Most discrete problems of size n have a simple solution that take time $O(n!)$: try all the $n!$ possible configurations of the input. The current state of computing technology is such so as to make this approach look hopelessly impractical. But with a molecular computer, this may be quite practical!

2.11.1. *Solving Hard Problems*

Consider a well known NP-complete problem, the directed Hamiltonian Path Problem (HPP): *Given a directed graph $G(V, E)$, with $|V| = n$, does there exist a path of length n-1 that covers every vertex, $v \in V$ exactly once?* We will show how to solve HPP in a reasonable time, using the DNA techniques that we have seen so far.

This involves the following steps:

1. Data → DNA-strands: Let a sequence of DNA nucleotides be represented as N, and its reverse complementary sequence is represented by N'. Let N_1N_2 denote the concatenation of the two sequences N_1 and N_2. Note that the reverse complement of N_1N_2 is $N'_2N'_1$.

   Each vertex, $v_i \in V$, is represented by a sequence $N_{1i}N_{2i}$. Every directed edge from v_i to v_j is represented by $N'_{2i}N'_{1j}$, and a directed edge from v_j to v_i by $N'_{2i}N'_{1j}$. It is easy to see that the edge-sequence can ligase with the vertex-sequences in a direction preserving manner.

   We construct short k-base DNA fragments representing each vertex and edge of the graph.
2. DNA Amplification & Ligation: The input data is amplified (see 2.9.2) and allowed to *ligase* or *renature*. This requires the right temperature and the right enzymes.
3. Sequence Testing**: We test the output in the following order:

*It is also believed that molecular computation is much more energy-efficient, that is, consumes much less energy than conventional computing

**It is easy to see that if a directed graph is modeled as above and many copies of nodes and edges is thrown in, and if there exists a Hamiltonian path, a double stranded sequence (molecule) whose length is the sum of the size of the nodes will be obtained

(a) *Right length*? Extract all those strands which have length kn using fragment fractionating by length (see 2.9.1).

(b) *Right path*? We have to check if any vertex appears more than once in this strand, extracted in step 1. This can be done by doing a *hybridization* (2.9.4) with each v_i, that is $N_{1i}N_{2i}$, in a sequence ensuring that there is exactly one such sequence.

Extract all those strands that have a single occurence of every vertex.

(c) *Verdict*. If at the end we are left with an empty test tube, then there is no HP, else the strand in the test tube defines the path.

The exact protocols to carry out the above have been identified and used to solve a HPP of a modest size of 7^{13}.

The reader may note that in principle all NP-complete problems could be solved in this manner. But what are the sources of problems in practice? The following issues are worth noting: (a) The increase in size of the problem is likely to increase the size of the data-DNA segments. This may require a very large amount, for example a 70-node HPP may require 425 gms of DNA (a human body has about 300 gms)! Further, carrying out the experiments may become cumbersome due to the sheer size. (b) Let us take the specific problem of the hamiltonian path. With the increase in the number of nodes (and edges), it is important to construct data-DNA sequences with no common patterns within or in concatenations. If sub-strings begin to match, then the molecules may pair up unexpectedly, or the testing of the sequences might fail.

3. A close look at some well-studied Tools

Molecular Biologists are seeking the help of computer scientists for assisting in the various kinds of problems they face. Reports of such joint efforts, or dealing with molecular biology problems, abound in literature. In the following sections, we will attempt to describe some of the tools employed, in the abstract, and, then its mapping to the molecular/structural biology problem.

3.1. *Intersection Graphs*

We will look at some basic definitions leading to interval graphs and some of its interesting properties.

Let $\mathcal{F}$ be a family of nonempty sets. The **intersection graph** of $\mathcal{F}$ is obtained by representing each set in $\mathcal{F}$ by a vertex and connecting two vertices by an edge if and only if their corresponding sets have a nonempty intersection.

3.1.1. *Interval Graphs*

The intersection graph of a family of intervals on a linearly ordered set* is called an **interval graph**. Alternately, an undirected graph G is called an *interval graph* if its vertices can be put into one-to-one correspondence with a set of intervals $\mathcal{I}$ of a linearly ordered set

*The real line, for instance.

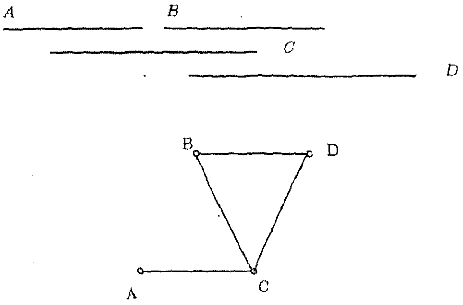

FIG. 11. Intervals on a line and the corresponding interval graph.

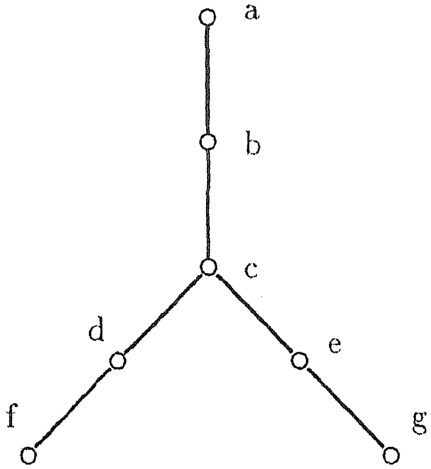

FIG. 12. A graph that is not an interval graph.

such that two vertices are connected by an edge of G if and only if their corresponding intervals have nonempty intersection. Why are we interested in interval graphs? If the DNA is considered a linear strand, the various clones correspond to the intervals and the graph gives a consistent linear ordering of the clones.

Figure 11 shows a set of intervals on a line and the corresponding interval graph. Inspired by the the structure of this graph, let us look at the following definitions.

A graph G is **triangulated** if every simple cycle of length strictly greater than 3 possesses a chord. This is also called the **chordal** graph.

The next natural question is: Are all interval graphs triangulated ? The answer is yes; but the converse is not true. Figure 12 shows a triangulated graph that is not an interval graph. This calls for further restrictions on a triangulated graph to satisfy the requirements of an interval graph.

An undirected graph G is called a *comparability graph* if each edge can be assigned a one-way direction in such a way that the resulting oriented graph (V, E) satisfies the following condition:

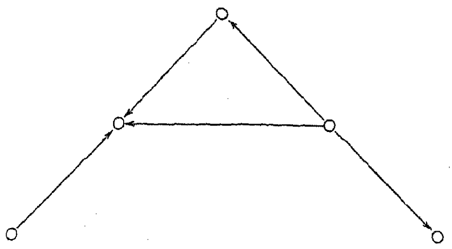

FIG. 13. A graph, and the orientation of the comparability graph.

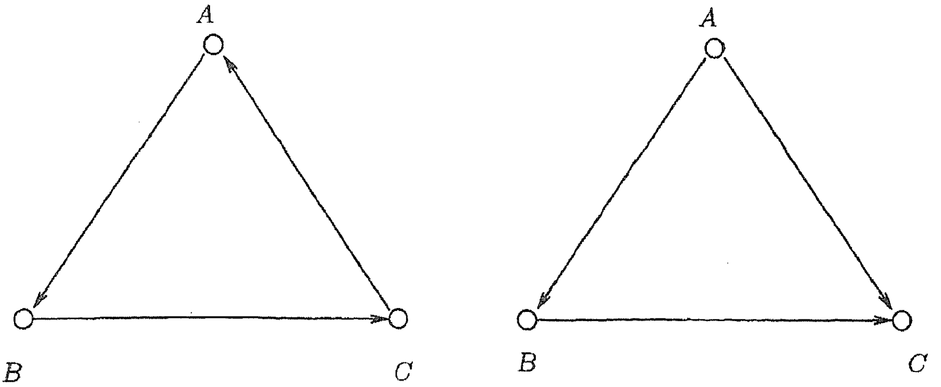

FIG. 14. An incorrect and a correct orientation of a comparability graph. [As an illustration let $A = \{2, 3\}$, $B = \{2, 3, 5\}$, $C = \{2, 3, 5, 7\}$ and the transitive relation is $\subset$ between the sets.]

$$ab \in E, bc \in E \Rightarrow ac \in E(\forall a, b, c \in V)$$

Informally, this graph captures a transitive relation (such as subset, for instance) of the elements of a set. See Figures 14 and 15 for some examples.

Let us define the maximal clique, since it has some interesting properties for interval graphs.

The maximal cliques, $M_1, M_2, \ldots, M_n$, of a graph G are the set of cliques that cover all the vertices of the graph and n is the smallest such possible number called the *clique cover number*, $k(G)$. Note that the largest value n can take is $|V|/2$ by considering each pair of vertices as a clique in a complete bipartite graph.

It may be interesting to note that $k(G)$ can be $|V|/2$ in a dense graph such as a complete bipartite graph, $K_{|V|/2,|V|/2}$ (with $[|V|/2]^2$ edges), and also in a sparse graph such as a path with $|V|$ vertices (with $|V|$-1 edges).

A **clique matrix, M**, is a matrix with rows corresponding to cliques and columns to vertices, and entry $[i, j]$ is 1 if clique M_i contains vertex v_j and 0 otherwise.

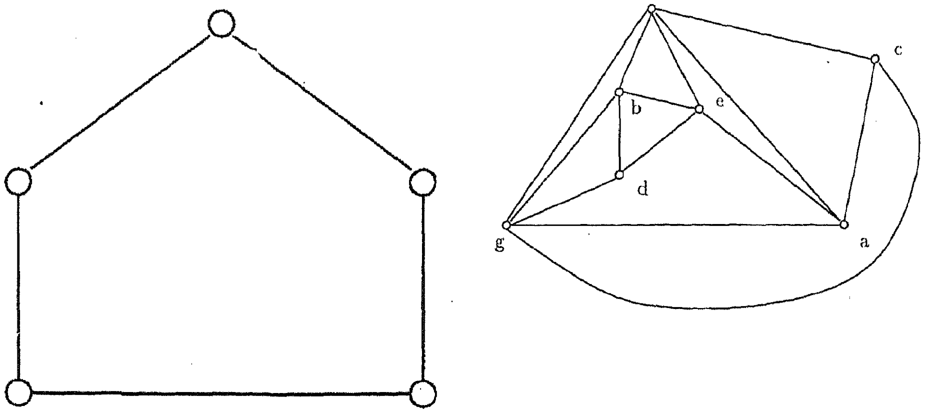

FIG. 15. Two graphs that are not comparability graphs. The graph on the bottom is the complement of the graph of Figure 12.

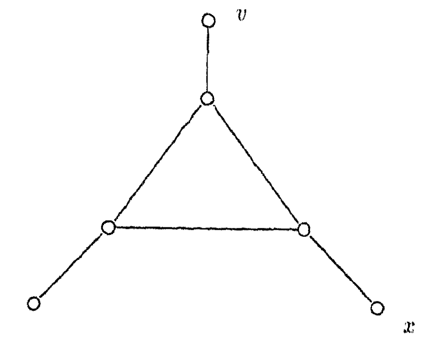

FIG. 16. v, w and x are astroidal triples.

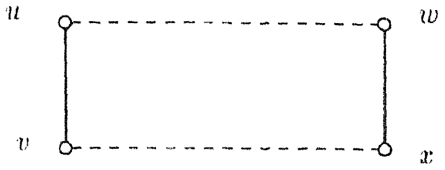

FIG. 17. uv and wx are edges in G and uw and vx in $\overline{G}$.

A matrix whose entries are are zeroes and ones, is said to have the **consecutive 1's property for columns** if its rows can be permuted in such a way that the 1's in each column occur consecutively.

A set of three vertices in an undirected graph G is said to be an **astroidal triple** if they are opairwise nonadjacent and any two of them is connected by a path which avoids the immediate neighbors of the third vertex. See Figure 16.

After the rather long list of definitions, we are now ready to look at some important characteristics of interval graphs.

Theorem 1: Let G be an undirected graph. The following statements are equivalent.

1. *G is an interval graph.*
2. *G is triangulated and its complement $\overline{G}$ is a comparability graph.*
3. *The maximal cliques of G can be linearly ordered such that, for every vertex x of G, the maximal cliques containing x occur consecutively.*
4. *G's clique matrix M has the consecutive 1's property for columns.*
5. *G is triangulated and contains no astroidal triples.*

Informal arguments for some of the proofs: $1 \Rightarrow 2$: It is easy to see that a chordless four or more cycle can not be an interval graph, since it violates the linear ordering*. Next, why must $\overline{G}$ be a comparability graph ? Let "to the left of" (or "to the right of") be the transitive relation. All the edges of $\overline{G}$ correspond to the pairwise intervals that do not intersect (else they would be in G). But, since G is an interval graph, "to the left of" is a consistent relation: hence the result. Figure 14 shows graphs that are not comparability graphs.

$2 \Leftrightarrow 3$: Let us look at two cliques A_1 and A_2 from the set of maximal cliques. There must exist some $x \in A_1$ and some $y \in A_2$, such that $xy \in \overline{G}$, for if it were not then $A_1 \bigcup A_2$ would be a maximal clique. Hence, every pair of cliques A_i and A_j must have at least one edge with each of its endpoints in A_i and A_j respectively. The next claim we make is that if there exist more than one such edge between cliques, they must all have the same orientation in the comparability graph of $\overline{G}$. See Figure 17. Assume u and v are distinct points in

*How can $a \prec b$, $b \prec c$, $c \prec d$, with $a \nprec c$, turn around and have $a \prec d$ ($\prec$ defines the linear order, say)?

A_i and w and x are distinct points in A_j. Of course, if they are not distinct the claim is trivially true. Further, both ux and vw cannot be in G (since, if uw, $vx \in \overline{G}$, we would have a chordless 4-cycle). Without loss of generality, let $ux \in \overline{G}$. Now, what orientation must ux have in the comparability graph of $\overline{G}$? It is easy to see that it is impossible to have uw and vx with different orientations. Thus we have shown there is a linear ordering of the maximal cliques (corresponding to the orientation of the comparability graph $\overline{G}$).

Next we have to show that a vertex x occurs in consecutive cliques. Assume the contrary: let A_1, A_2, A_3 be cliques in that linear order and $x \in A_1$, $x \in A_3$, but $x \notin zA_2$. Let $y \in A_2$, where, $y \neq x$. Then the orientation of xy $(x \in A_1, y \in A_2)$ and yx $(y \in A_2, x \in A_3)$ give a contradiction.

1 $\Leftarrow$ 3: For each vertex $v \in V$, let $I(x)$ denote the set of all maximal cliques of G which contain x. To show $xy \in E \Leftrightarrow I(x) \bigcap I(y) \neq \Phi$.But $xy \in E \Leftrightarrow$ both the vertices are in the same maximal clique.

3 $\Leftrightarrow$ 4: By definition (4 is a paraphrasing of 3 or vice-versa).

1 $\Leftrightarrow$ 5: This can be seen by studying Figure 16. Note that it is impossible to have intervals corresponding to these vertices just as in Figure 12.

For the remainder proofs and more formal version of the arguments here, the reader is advised to look at[2].

3.1.2. *Data Structure – PQ Trees*

The general consecutive arrangement problem is the following: *Given a finite set X and a collection $\mathcal{I}$ of subsets of X, does there exist a permutation π of X in which the members of each subset $I \in I$ appear as a consecutive subsequence of π?* PQ-tree is a data structure that solves this problem very efficiently in linear time. What is the relation to interval graphs ? In the interval graph problem, X is the set of maximal cliques and $\mathcal{I} = \{I(v)\}_{v \in V}$, where $I(v)$ is the set of all maximal cliques containing v. The consecutive arrangement and consecutive 1's problems are equivalent which is the interval graph problem.

Let us look at the details of this data structure in the context of the consecutive arrangement problem. Let X be a finite set and $\mathcal{I}$ a collection of subsets of X. A PQ-tree is a rooted tree whose internal nodes are of two types: P and Q. The children of a P-node occur in no particular order while those of a Q-node appear in a left to right or right to left order. We designate a P-node by a circle and a Q-node by a rectangle. The leaves of T are labeled bijectively by the elements of the X. In other words, we do not permit more than one leaf to have the same label. What happens if we do? In that case, we have the *shortest common superstring* problem which is defined as follows: *Given a finite set X and a collection $\mathcal{I}$ of subsets of X, what is the shortest string in which the members of each subset $I \in \mathcal{I}$ appear as a consecutive subsequence?* This problem is known to be NP-hard[16].

The *frontier* of a tree T is the permutation of X obtained by reading the labels of the leaves from left to right. Two PQ-trees T and T' are equivalent, denoted $T \equiv T'$, if one can be obtained from the other by applying a sequence of the following transformation rules:

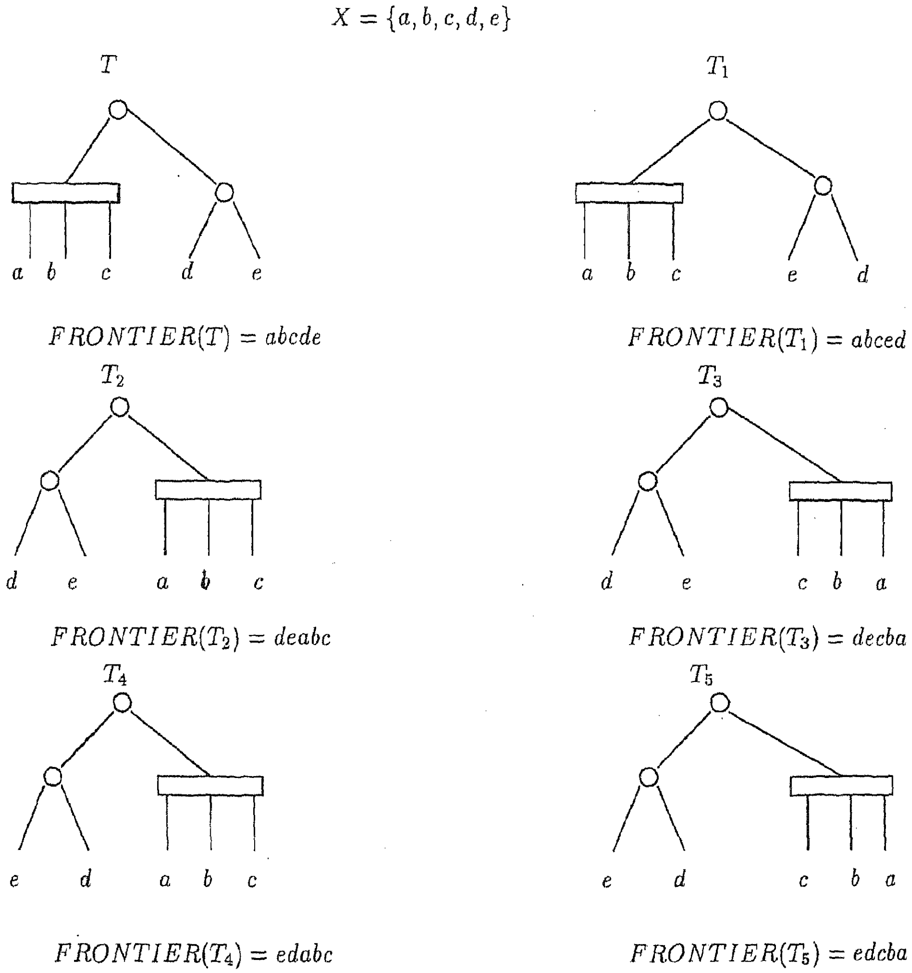

FIG. 18. A tree T, with $T_1, T_2, \ldots, T_5$ equivalent to it. Note that CONSISTENT(T) = {*abcde*, *abced*, *deabc*, *decba*, *edabc*, *edcba*}.

1. Arbitrarily permute the children of P-node.
2. Reverse the children of a Q-node.

Any frontier obtainable from a tree equivalent with T is said to be *consistent* with T, and we define

$$\text{CONSISTENT}(T) = \{\text{FRONTIER}\}(T') \mid T' \equiv T\}.$$

See Figure 18 for an example.

The classes of consistent permutations of PQ-trees over a finite set X form a lattice. The null tree, T_O, has no nodes and CONSISTENT(T_O) = π. The *universal tree*, T_U, has one internal P-node, the *root*, and a leaf for every member of X (Figure 19). Note that

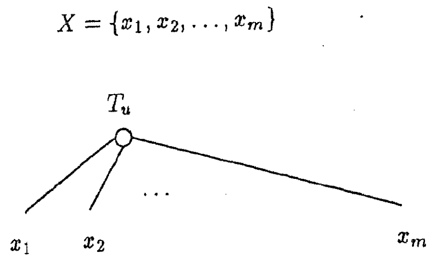

FIG. 19. The Universal tree, T_U.

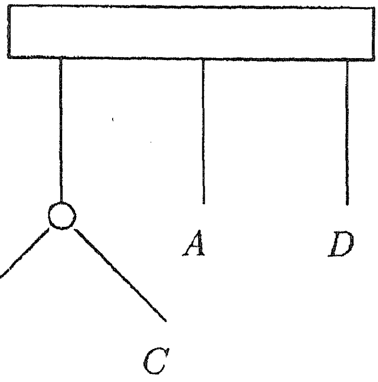

FIG. 20. The tree corresponding to $\Pi(I)$ where $I = \{\{A, B, C\}, \{A, D\}\}$.

$$\text{CONSISTENT}(T_U) = \text{all possible permutations of } X.$$

Let $\mathcal{I}$ be a collection of subsets of a set X, and let $\Pi(\mathcal{I})$ denote a collection of all permutations π of X such that the members of each subset $I \in \mathcal{I}$ occur consecutively in *pi*. For example, if $\mathcal{I} = \{\{A, B, C\}, \{A, D\}\}$, then $\Pi(\mathcal{I}) = [DABC], [DACB], [CBAD], [BCAD]\}$. Figure 20 shows the corresponding PQ tree.

Let us note, without proof, two important facts about PQ-trees.

- For every collection of subsets $\mathcal{I}$ of X there exists a PQ-tree T such that $\Pi(\mathcal{I}) =$ CONSISTENT(T).
- For every PQ-tree T there exists a collection of subsets $\mathcal{I}$ such that $\Pi(\mathcal{I}) =$ CONSISTENT(T).

There is one important pattern matching algorithm called REDUCE on the PQ-trees. The algorithm is briefly sketched here, for details refer to[1].

Input: A PQ-tree T and a set I.

Output: A PQ-tree T' such that CONSISTENT(T') = CONSISTENT(T) $\cap$ Set of all permutations where the elements of I are consecutive.

Essence of the algorithm: The algorithm looks at the elements of I and looks for a specific pattern in the tree T modifying it locally, by looking at 2 (or fewer) levels at a time, in a bottom-up fashion. The authors call this the template matching process: they have shown that 11 templates are sufficient to accommodate all possible configurations. The local modifications are repeated until all the requirements are met or an impossible situation is obtained. The former reports a success with T′ and the latter a failure. It is important to note that with a clever pre-processing of the tree T, the algorithm can compute T' in linear time. Figures 21 and 22 illustrate the algorithm.

The description of the last paragraph really gives no insight into why (or how) the algorithm works. The most intriguing part seems to be the templates; let's see how a small set of 11 patterns can capture all that is required in the pattern matching process.

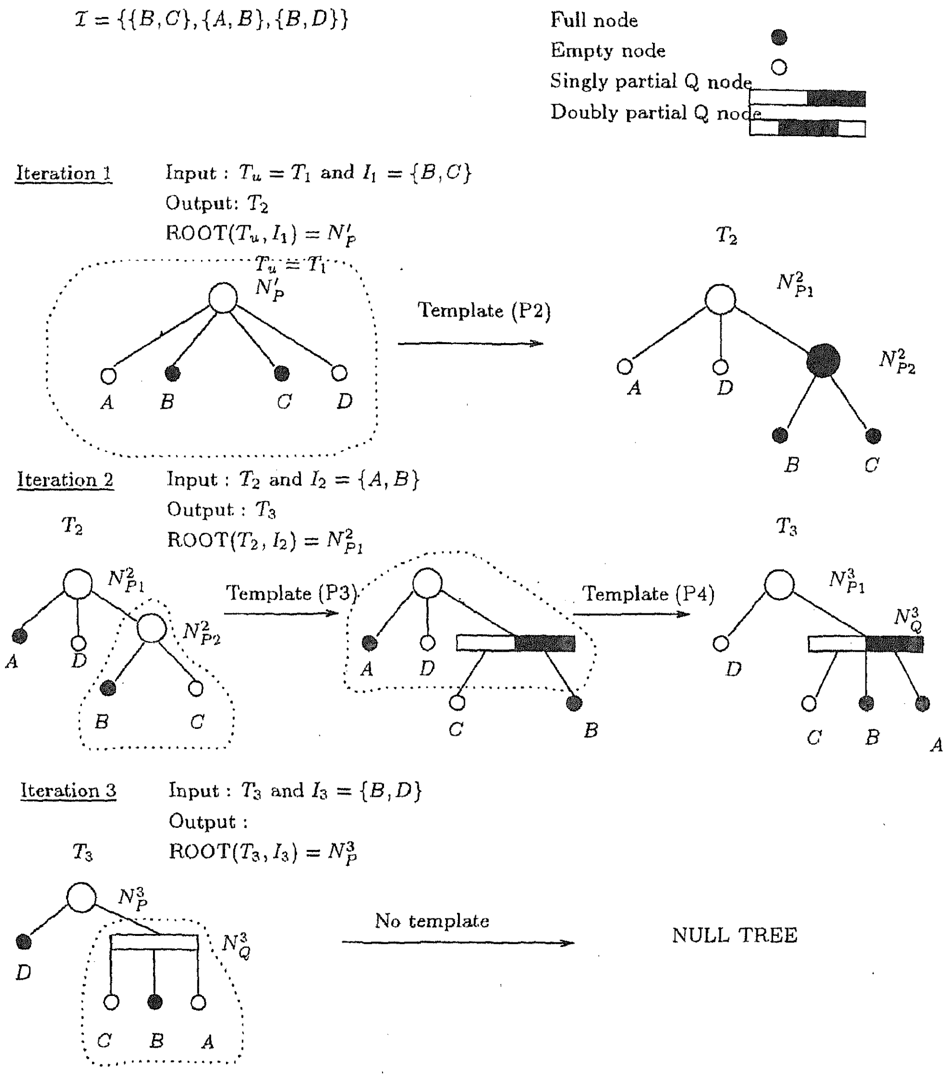

FIG. 21. An example showing the REDUCTION process.

The algorithm works in two phases:

Phase 1: Mark the nodes.

1. A leaf node is marked *full* or *empty* depending on whether it belongs to I or not.
2. A non-leaf node is marked *full*, *empty* or *partial* depending on whether its children are all marked *full*, *empty* or otherwise. Marking a P node *full* or *empty* is equivalent to the use of template P0 and similarly Q0.

Phase 2: Moving bottom-up, replace certain structures (templates) by appropriate patterns of the nodes. Define ROOT(T, I) as the root of the smallest subtree in T such that I is

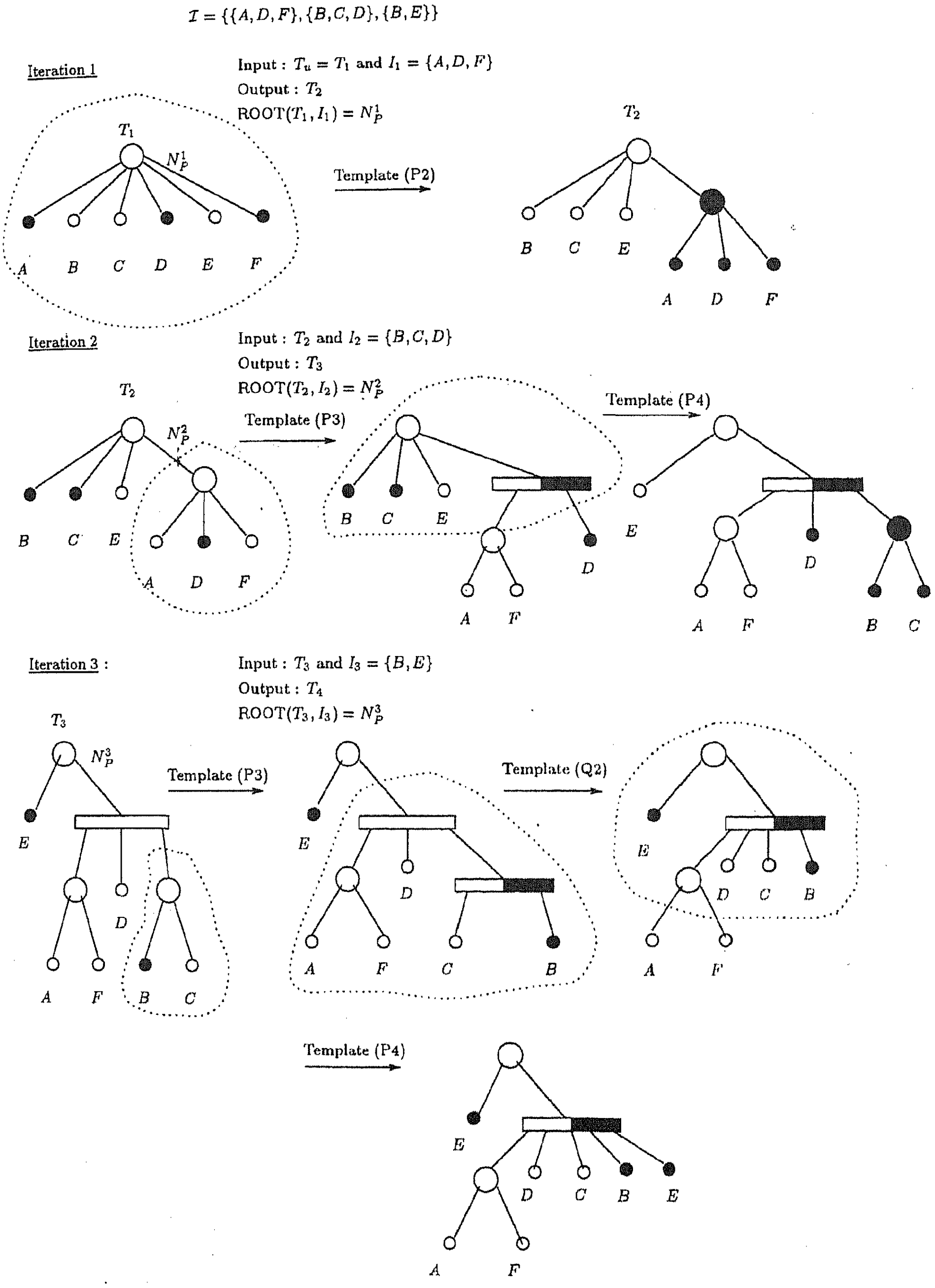

FIG. 22. Another example demonstrating the use of the templates.

a subset of the descendents. Further, a Q node can be *singly-partial* if it has all *full* nodes to the left and all *empty nodes* to the right or vice-versa. A Q node can be *doubly-partial* if it has a sequence of *empty* nodes followed by *full* nodes followed by *empty* nodes.

The templates are of the following kinds:

1. The root of the template is a P node.
    a) All its children are full (template P1).
       Replacement: The P node is marked full.
    b) Not all its children are full.
        i. The P node is ROOT(T, I).
            A. The P node has *no* partial children (template P2).
               Replacement: Create a P node as the parent of all the *full* children and make this new node the child of the P node.
            B. The P node has *one* partial child (template P4).
               Replacement: Create a *full* P node which is the parent of all the *full* children. This new node is made the child of the partial node at the *full* end (so that all the *full* children are adjacent).
        ii. The P node is not ROOT(T, I).
            A. The P node has no *partial* children (template P3).
               Replacement: Create two P nodes, one *empty* and the other *full*, with the former as the parent of all the *empty* children and the latter that of all the *full* children. Replace the P node by a *singly-partial* Q node which is made the parent of the two new P nodes.
            B. The P node has one *partial* child (template P5).
               Replacement: Create two P nodes, one *empty* and the other *full*, with the former as the parent of all the *empty* children and the latter that of all the *full* children. Make these two new P nodes the siblings of the partial child (*empty* P node at the end adjacent to empty children, and similarly, the *full* children). Replace the P node by a *singly-partial* Q node.
            C. The P node has two *partial* children (template P6).
               Replacement: Create a *full* P node as the parent of all the *full* children. The two *partial children* (which are necessarily Q nodes) are merged into one *doubly-partial* Q node with the new *full* P node next to the other *full* children.
2. The root of the template is a Q node.
    (a) All its children are full (template Q1).
        Replacement: The P node is marked full.
    (b) Not all its children are full.
        i. The Q node has one *partial* child (template Q2).

Replacement: Make all the children of the *partial* child, the children of the Q node in an appropriate order (maintaining adjacency). The Q node is labeled *singly partial*.

ii. The Q node has two *partial* children (template Q3).
Replacement: The Q node is made *doubly partial*, with all the children of the *partial* children becoming the children of the Q node in an appropriate sequence (maintaining adjacency).

The reader may note that the linear ordering enforces only 2 possible situations of 1 or 2 partial children, with all other configurations being equivalent to one of the two. This helps keep the number of templates down. It is an interesting exercise to informally verify that these templates are sufficient.

See Figures 21 and 22 for applications of the templates to two examples.

Possible Extension: It may be worthwhile to explore if this idea can be extended to carry out pattern matching in 2 dimension and if a small number of such templates can be identified.

3.1.3. *Algorithms for Interval Graphs*

Let us define a few more characteristics of graphs and see how easily they can be computed for the graphs under study.

$\omega(G)$ is the number of vertices in the maximum clique of G; it is called the **clique number** of G.

A **stable set** (or an **independent set**) is a subset of the vertices no two of which are adjacent. $\alpha(G)$, the **stability number** of G, is the size of the largest possible *stable set*.

$\chi(G)$ is the minimum number of colors required to color the vertices of G such that no two adjacent vertices have the same color, called the **chromatic number**.

Note the duality of the above notions,

$$\omega(G) = \alpha(\overline{G}) \text{ and } \chi(G) = k(\overline{G}).$$

Recall that $k(G)$ is the **clique cover number**.

Theorem 2: *For an interval graph G, $\chi(G)$, $\omega(G)$, $\alpha(G)$ and $k(G)$, with the maximal cliques, can be computed in linear time.*

Informal argument This is going to be a mere sketch of the arguments. For details see [2]. Let us accept the following without proof:

G is triangulated $\Leftrightarrow$ G has a perfect vertex elimination scheme, $\sigma = [v_1, v_2, \ldots, v_n]$. The *perfect vertex elimination scheme* is an ordering of the vertices of G such that $X_i = \{v_j \in \text{Adjacent}(v_i) | j > i\}$ is complete $\forall i$. See Figure 23.

$$\chi = \max_i\{1 + |X_i|\},\ i = 1, 2, \ldots, n.$$

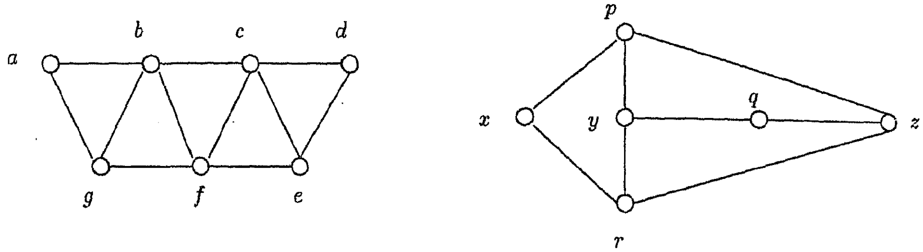

FIG. 23. $\sigma = [a, g, b, f, c, e, d]$ is a perfect vertex elimination scheme, not necessarily unique. The second graph, on the other hand, has none.

On σ, define a sequence $y_1, y_2, \ldots, y_t$, for some t, as follows:

$$y_1 = \sigma(1).$$

y_i is the first vertex that follows y_{i-1} and $y_i \notin X_{y1} \cup X_{y2} \cup \ldots \cup X_{yi-1}$. The set $\{y_1, y_2, \ldots, y_t\}$ is a maximum stable set of G and $Y_i = \{y_i\} \cup X_{yi}$ ($i = 1, 2, \ldots, t$) comprises the minimum clique cover of G.

$\alpha(G) = t$.

Theorem 3. An interval graph can be recognized in linear time.

Informal argument An interval graph is necessarily triangulated. It takes linear time to compute the set of maximal cliques of a triangulated graph. The consecutive one's property in clique matrix **M** is equivalent to the consecutive arrangement which can be computed in linear time.

3.1.4. *Proper Interval Graphs*

If the primary goal is obtaining the linear ordering of "intervals", the ones that are fully contained in others shed no new light, and, removing them might be useful. Let us study such a subclass of interval graphs called the **proper interval graphs**.

Before studying an interesting theorem, let us look at a definition. A real-valued function $u : V \rightarrow \Re\}$ is called a **semiorder function** for a binary relation (V, P) if the following condition is satisfied:

$$xy \in P \Leftrightarrow u(x) \geq u(y) + \delta \, \forall x, y \in V \text{ and some } \delta > 0.$$

Theorem 4: *Let $G = (V, E)$ be an undirected graph. The following are equivalent:*

1. *$\overline{G}$ is a comparability graph and every transitive orientation of $\overline{G} = (V, \overline{E})$ is a* **semiorder**.
2. *G is an interval graph containing no induced copy of $K_{1,3}$*.*
3. *G is a proper interval graph.*
4. *G is a unit interval graph, i.e., all intervals are of unit length.*

*A complete bipartite graph with 1 and 3 vertices in the partition.

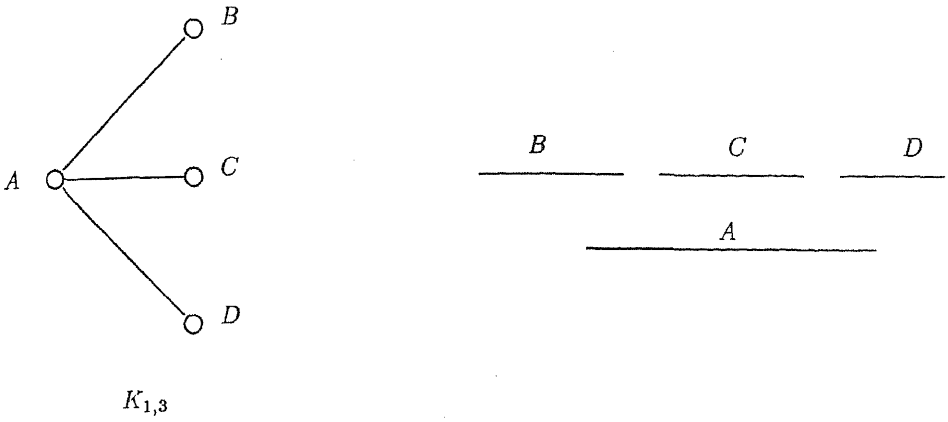

FIG. 24. The only possible layout of the intervals corresponding to $K_{1,3}$ are shown on the right (B, C, D can be placed in any order).

Informal argument 3 ⇔ 2: See Figure 24.

See [2] for the details of the rest of the proof.

3.1.5. *Circular-arc graphs*

The intersection graphs obtained from collections of arcs on a circle are called **circular-arc graphs**. This kind of interval graphs are of interest to us, since some DNA strands like that of *E. Coli* are circular.

See Figure 25 for an example. Note that a circular-arc graph is not even *triangulated*. Further, it is not a *perfect* graph. A graph G is *perfect* if and only if $\chi(G) = \omega(G)$. Note that an interval graph is *perfect* since it is necessarily triangulated (and, all triangulated graphs are *perfect*).

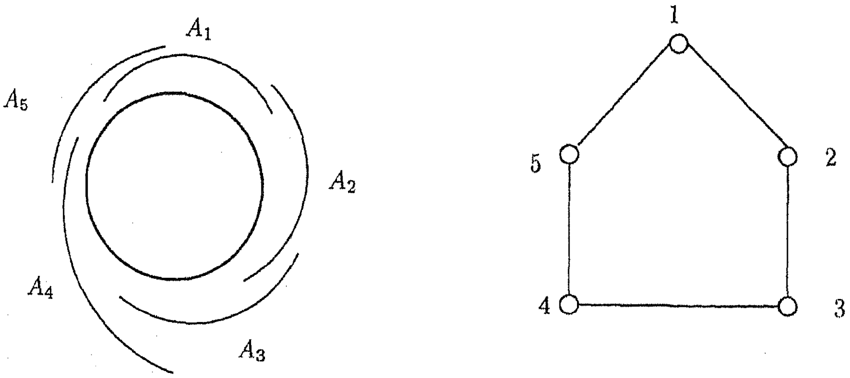

FIG. 25. The circular arc representation of the **circular-arc** graph on the right.

We look at some results without proofs:

Theorem 5: An undirected graph $G = (V, E)$ is a circular-arc graph if and only if its vertices can be (circularly) indexed $v_1, v_2,..., v_n$ so that $\forall$i and j,

$$v_iv_j \in E \Rightarrow \begin{cases} \text{either } v_{i+1},\dots,v_j \in Adj(v_i) \\ \text{or } v_{j+1},\dots,v_i \in Adj(v_j) \end{cases}.$$

If $i > j$, then $v_{j+1},..., v_i$ means $v_{j+1},..., v_n, v_1,..., v_i$.

Theorem 6 *An undirected graph G is a circular-arc graph if its augmented adjacency matrix (adjacency matrix by adding 1's along the main diagonal) has the circular 1's property for columns.*

3.2. *Dynamic Programming*

Divide-and-conquer solves problems by dividing the problem into independent subproblems and then combining the solutions. If the subproblems are not independent, *i.e.*, they have some overlapping subsubproblems, dynamic programming is applicable. In this method, the solution to the common subsubproblem is computed and stored in a table, called **memoization**, and subsequent computations refer to this computed value.

A problem exhibits **optimal substructure** if an optimal solution to the problem contains within it optimal solutions to subproblems. We illustrate the dynamic programming method by giving examples where it is applicable.

The Longest Common Subsequence (LCS) Problem

Given two sequences $X = \langle x_1, x_2,..., x_m\rangle$ and $Y = \langle y_1, y_2,..., y_n$, the problem is to find $Z = \langle z_1, z_2,..., z_l\rangle$ with largest l where $z_1 = x_{m_1} = y_{n_1}, z_2 = x_{m_2} = y_{n_2}, \dots, z_l = x_{m_l} = y_{n_l}$, with $m_1 <$ m2 $<...< m_l$, and $n_1 < n_2 <...< n_l$.

It is easy to see that this satisfies the optimal subproblem property. The following recursive formula is obtained.

$$\text{Len}[i,j] = \begin{cases} 0 & i = 0 \text{ or } j = 0 \\ \text{Len}[i-1, j-1] + 1 & x_i = y_j \text{ and } i > 0, j > 0 \\ \max(\text{Len}[i-1, j], \text{Len}[i, j-1]) & x_i \neq y_j \text{ and } i > 0, j > 0 \end{cases}$$

Len[i, j] denotes the length of the longest common subsequence of $x_1, x_2,..., x_i$ and $\langle y_1, y_2,..., y_j\rangle$.

Clearly, the running time of this algorithm is $O(mn)$. Further, if the values are stored in an $m \times n$ table, the longest common subsequence may be generated, from this table, in $O(m + n)$ time.

Edit Distance of Strings

Given two sequences $X = \langle x_1, x_2, \ldots, x_m\rangle$ and $Y = \langle y_1, y_2, \ldots, y_n\rangle$, and a set of permissible edit operations (like insert, delete, twiddle etc) with costs, the edit distance is the minimum cost of transforming X to Y using a sequence of editings.

Note that the cost of every edit operation need not be the same, thus indicating a preference of some edit operations or the probability of such a transformation occuring if the strings have been obtained from experiments or nature (like the DNA strings).

Let us make a very general definition of the edit operations and give specific instances later. Let the total number of edit operations allowed be R with each edit operation E_r costing e_r that uses K_r characters from the first string and L_r characters from the second string*. It is easy to obtain the following recursive formula.

$$C[i,j] = \begin{cases} 0 & i = 0 \text{ or } j = 0 \\ C[i-1, j-1] + 1 & x_i = y_j \text{ and } i > 0, j > 0 \\ \min\left(C[i-K_r, j-L_r] + e_r\right) & x_i \neq y_j \text{ and } i > 0, j > 0 \\ & \text{where } E_r \text{ is applicable} \end{cases}$$

$C[i, j]$ denotes the minimum cost of edit-transforming $\langle x_1, x_2, \ldots, x_i\rangle$ to $\langle y_1, y_2, \ldots, y_j\rangle$.

Of course, the burden is on finding effective E_r with meaningful e_r. Specific definitions of E_r give rise to otherwise well known problems in practice. Below, we list a set of well known problems[17] whose solution are built around the basic *edit distance* problem shown above.

- Fitting one sequence into another:

  Here the assumption is that $m << n$, although this does not affect the mathematical formulation. Figure 26 shows the situation where the smaller X is fitted in the longer Y. The formulation stays the same except for an additional index l that moves over Y denoting the starting point of matching x_1 at y_1. Let $C^l[i, j]$ denote the minimum edit distance between $\langle x_1, x_2, \ldots, x_i\rangle$ and $\langle y_l, y_{l+1}, \ldots, y_j\rangle$. The following modifications are called for:

  1. Let $C^l[i, j] = C[i, j]$ with the added restriction $C^l[i, j] = C[i, j] = 0$, when, $i < l$.
  2. Compute $C^l[i, j]$ for $l = 1, 2, \ldots, n$.
  3. Best-fit-distance = $\min_{l,j,\, l<j} \{C^l[\text{m,j}]\}$.

  The time taken is $O(mn^2)$.

- **Local Alignment & Clumps:**

  See Figure 26. Here we are looking for similar segments $x_k, x_{k+1}, \ldots, x_i$ and $y_l, y_{l+1}, \ldots, y_j$. As before, we use the same formulation as that of *edit distance* except for two more indices l and k. l moves over Y and k moves over X. Let $C^{kl}[i, j]$ denote the

*For example we may define E_1 to be the edit operation of replacing 3 characters by 2 characters: then $K_1 = 3$ and $L_1 = 2$.

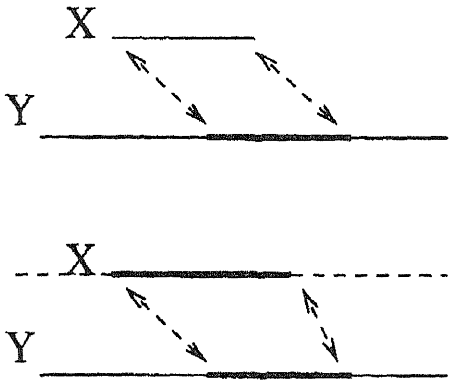

FIG. 26. Fitting a sequence into another. (2) The local alignment problem.

minimum edit distance between $\langle x_k, x_{k+1},\ldots, x_i\rangle$ and $\langle y_l, y_{l+1},\ldots, y_j\rangle$. The following modifications are called for:

1. Let $C^{kl}[i, j] = C[i, j]$ with the added restriction Ckl[i, j] = C[i, j] = 0 when $i < k$ and, $j < l$.
2. Compute $C^{kl}[i, j]$ for $k = 1, 2,\ldots, m$, $l = 1, 2,\ldots, n$.
3. Best-fit-distance = $\min_{i, j, k, l, k < i, l < j}\{C^{kl}[i, j]\}$.

The time taken is $O(m^2n^2)$.

- Self Comparison:
  The task is to find repeated non-overlapping sequences in a (single) string. We can treat it exactly like the *local alignment and clumps problem* where $X = Y$ with the following restriction: $C^{kl}[i, j] = C[i, j] = 0$ when $i < k$, $j < l$, and, $l < i$. This makes sure that we are always comparing a substring that appears strictly to the left of the other substring thus ensuring non-overlap.

- (Non-intersecting) Inversion:

  For our purposes, this is the same as the *edit distance* problem with inversion as one of the edit operations. Inversion is placement of a substring x_k, $x_{k+1},\ldots$, x_i as $\langle \bar{x}_i, \bar{x}_{i-1},\ldots,\bar{x}_k\rangle$*.

- **Map Alignments:**

  Let us define the problem in abstract. We are given two strings, as before, X and Y. Along the strings there are marked points each denoting a pair of information: the position, an integer, and an attribute. The task is to get a best correspondence between the marked points in the two strings.

*In the language of DNA nucleotides $\bar{A} = T, \bar{T} = A, \bar{C} = G, \bar{G} = C$.

This can be treated as the general *edit distance* problem with an appropriate distance function defined over the tuples and a set of acceptable edit operations like replace, delete etc. The distance between unmarked points is defined to be zero.

We have given the most naive formulation for the above problems. In practice, there can be ways of making them work much better by using restrictions that apply to the problems at hand.

Dynamic Program seems to be a pancea for most problems in practice. But, the user must proceed with caution: there are lots of problems out there (which have not yet been proved to be NP-complete/NP-hard) which can not be conquered by this otherwise very effective tool.

3.3. *Model-based Matching (Geometric Hashing)*

Object recognition is a major task in computer vision. Geometric Hashing[18] is a technique for recognizing objects in a cluttered scene, and, works particularly well for rigid objects. We wish to use ideas from this in the *protein docking* or the *molecular recognition* problem.

There are two aspects to a recognition system:

1. Models: These are objects the system must potentially recognize. The system is familiarized with these objects in various ways depending on how the recognition system works. Analogy: In molecular recognition, the models would be short DNA sequences or small molecules such as an enzyme inhibitor, the *ligand*.
2. Scene: The system is presented with the scene in which it is to locate the objects, whose models it is acquainted with.

   Analogy: In molecular recognition, this could be the DNA or the protein, the receptor.

An object may look very different in different scenes, depending on the view (from what angle and what distance it is being viewed, for instance) or on the lighting (was it captured during the day, twilight, evening?) and various other factors. So, the basic question is: what is it that defines the object unambiguously? This is a hard question to answer. Geometric Hashing technique takes the view that an object is defined by a collection of *interest points* and their relative positions*. These object-models are indexed in a transformation invariant way. In order to recognize partly occluded objects sufficient redundancy is allowed.

The following steps capture the geometric hashing process:

1. Model Inversion: Let us first look at what the input and output are and then describe how the inversion is carried out.

   Input: M objects with N models each (*i.e.*, a total of MN object models). We could assume that for each of these MN object-models, we have extracted K *interest points* (with attributes if any**). In other words, the input to the model inversion

*These points can be high curvature points or junction points.

**The attribute could be the curvature value.

process is the 2D coordinates of the K interest points of each of the MN object-models.

Output: A two dimensional hash table, $\mathcal{H}$, with each entry carrying the following information:

- a pointer to a (model-object, integer) pair. This integer corresponds to a basis, which we discuss in the next step.
- a weight reflecting the confidence with which that association is made. It is a real number between 0 and 1.

We need to carry out a transformation invariant indexing for each of the object-models i, $i = 1, 2, \ldots, MN$. Define a new coordinate system for every three non-collinear points of an object-model (called the basis): there are at most $P = 6$ such possibilities. For each possibility p, $p = 1, 2, \ldots, P$, for all the interest points $\left(x_k^{ip}, y_k^{ip}\right)$, $k = 1, 2, \ldots, K$, transform them to the new coordinates, $\left(x_k^{\prime ip}, y_k^{\prime ip}\right)$ $k = 1, 2, \ldots, K$. Now, $\mathcal{H}\left[\text{round}\left(x_k^{\prime ip}\right), \text{round}\left(y_k^{\prime ip}\right)\right] = (i, p)$. Recall that i is the pointer to the object model under consideration, and, p the pointer to the basis.

What is the size of $\mathcal{H}$? Let the range of the first index be $x_{\text{low}} \ldots x_{\text{high}}$ and similarly that of second index be $y_{\text{low}} \ldots y_{\text{high}}$. Then,

$$x_{\text{low}} = \min_{i,p,k}\left(\text{round}\left(x_k^{\prime ip}\right)\right), y_{\text{low}} = \min_{i,p,k}\left(\text{round}\left(y_k^{\prime ip}\right)\right)$$

$$x_{\text{high}} = \max_{i,p,k}\left(\text{round}\left(x_k^{\prime ip}\right)\right), y_{\text{high}} = \max_{i,p,k}\left(\text{round}\left(y_k^{\prime ip}\right)\right).$$

Uncertainity Factor: An uncertainty window with Gaussian distribution is built around every entry in the table. This is the weight that we store with every entry in $\mathcal{H}$. In other words, the immediate neighbours around the entries of $\mathcal{H}$ in the last step, also point to the same (object-model,basis) pair but with lower weight.

2. Scene Inversion: This step is exactly like that of the model inversion except that there is only 1 scene (unlike MN models). We construct the hash table, $\mathcal{T}$, in a similar fashion with the following exception: there are no model pointers since there is only one scene and no uncertainity window is computed. Note that $\mathcal{T}$ is only a notion and in practice the table may not be explicitly created.
   Assume L interest points are collected from the scene where $L >> K$.

3. Matching: This is done separately for every basis of the scene. We maintain scores for each (object-model,basis) pair of step *1*: this is initialized to zeroes for all entries. Let $\mathcal{T}[x, y]$ point to the basis being considered. Now, if $\mathcal{H}[x, y]$ is non-empty, then we add the weight of this entry to the score of the (object-model, basis) pair of this entry.

   For each basis, we obtain a table of scores of the (object-model, basis) pairs.

4. Verdict: Now, we are ready to give the verdict of a "hit" or "no-hit". We pick up the model that has been hit most over all the bases of the scene a sufficient number

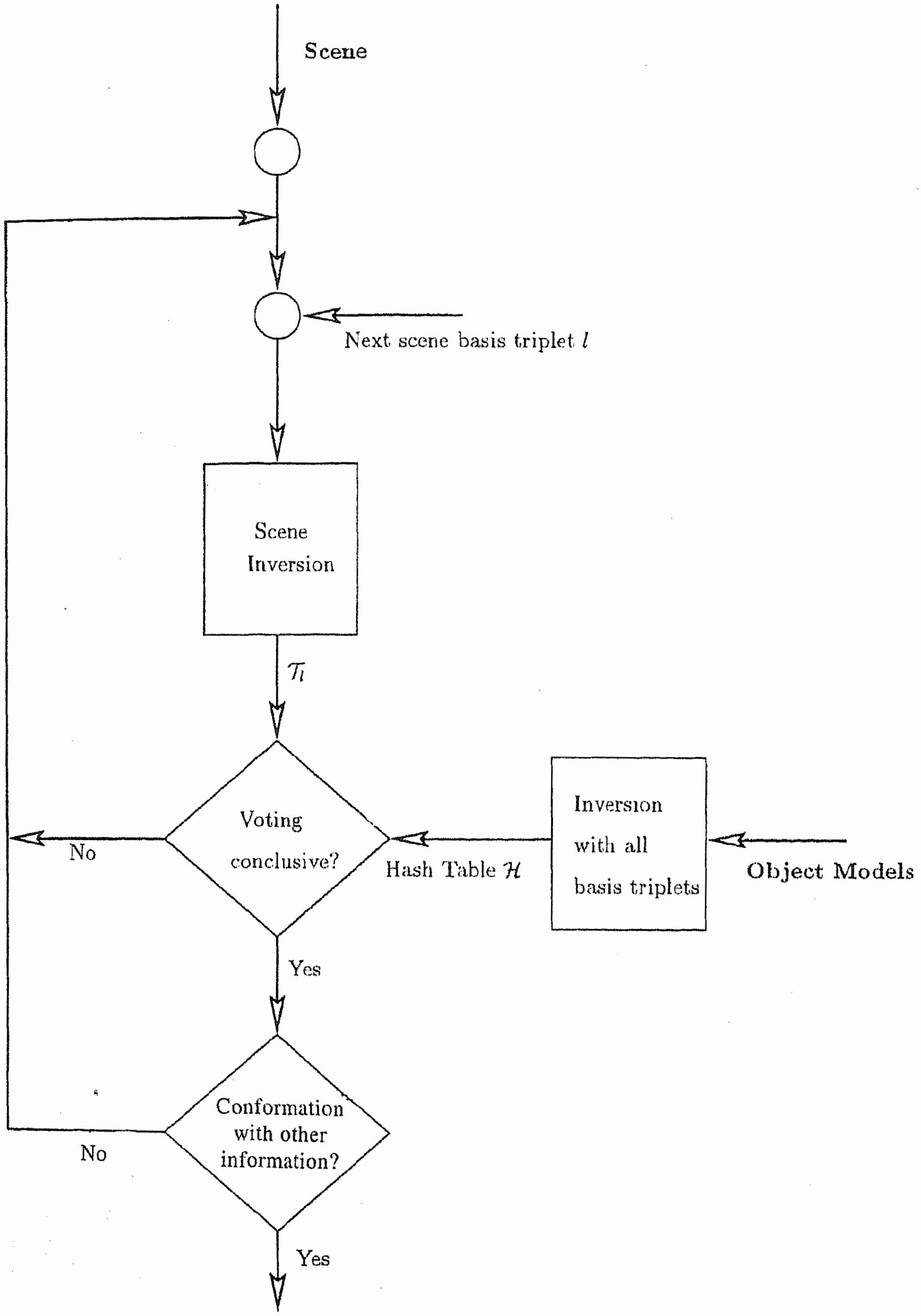

FIG. 27. The steps in the geometric hashing process.

Tool	*Problems*	*References*
Intersection Graphs	Sequencing by hybridization (SBH)	See [23]
	SBH using k-tube probes: Given 4^l indicator values, 0 or 1, each denoting the absence/presence of one of the k-consecutive patterns over the alphabet A, C, G, T, the task is to determine the unique super-string that is consistent with the data.	
Interval Graphs, & PQ-trees	*Partial Digest & Double Digest Problem*	See [4].
	Partial Digest Problem: Given a set of N real values $l_1, l_2, \ldots, l_N$, where possibly $l_i = l_j$, $i \neq j$, the task is to obtain a rel $P > 0$, with an ordered sequence $p_0 = 0 < p_1 < p_2 < \ldots < p_M = P$ such that $\forall_I$, $l_i = \lvert p_k - p_j \rvert$, for some $k > j$ and P is the minimum such value. Double Digest Problem: Given three sets of real values $a_1, a_2, \ldots, a_N$ (possibly $a_i = a_j$, $i \neq j$), $b_1, b_2, \ldots, b_N$ (possibly $b_i = b_j$, $i \neq j$), and, $c_1, c_2, \ldots, c_N$ (possibly $c_i = c_j$, $i \neq j$), the task is to obtain a real $P > 0$, with an ordered sequence $p_0 = 0 < p_1 < p_2 < \ldots < p_M = P$ such that $\forall I$, $a_i = \lvert p_k - p_j \rvert$, for some $k > j$, similarly for b_i, c_i and P is the minimum such value. Remarks: These problems arise while constructing the physical map of clones from Gel Electrophoresis (see section2 .9) technique which gives the length/mass of each segment when cleaved with a restriction enzyme (see section 2.9.1). IN the double digest problem two restriction enzymes are used, to give sequences a_i and b_i. c_i is obtained by using both the restriction enzymes. The single digest problem can possibly give many plausible solutions, double digest attempts to pin down the right one.	
Dynamic Programming	Evolutionary Tree Comparison	See [10]
	Maximum Agreement Subtree Problem: Given a set A of two rooted trees T_1 and T_2 leaf-labeled by the elements of A, the task is to find a maximum cardinality subset B of A such that the restriction of T_1 and T_2 to B are topologically isomorphic.	
Alternating Cycles & Edge-colored Graphs	Genome Rearrangement Problem	See [11, [12]
	Genome Rearrangement Problem: Given two sequences (or genomes), the task is to find the shortest sequence of transformations that take one genome to the other. The transformations can be deletions, reversals, additions etc., which helps understand the evolution process.	
Algebraic Techniques, Geometric Matching & Geometric Hashing	Protein folding Protein Docking problems.	See [6]. See [6].
	Protein Docking: Here a protein, the *receptor*, is matched with another smaller protein, a DNA sequence or an enzyme inhibitor, called the *ligand*. The goal is to determine if the two can associate. Essentially two problems must be addressed: first is the geometric problem of obtaining a feasible configuration and the second is the chemical problem of evaluating the energies (see section 2.4.1)	
Algebraic Techniques (Inverse Kinematics)	Protein folding Ring Closure	 See [6].
	Protein folding: See section 2.4.1 Ring Closure: The problem is to compute conformations of a molecule with a cyclic structure in which the constraints imposed by the bond lengths and angles are respected.	
Motion Planning (Robotics)	Protein Docking	See [25]
Numerical Optimization	Protein folding	See Section 2.4.1. Also see [21].
Computational Linguistics	Understanding genetic sequences	See [22].
Text Compression Techniques	DNA Classification	
	DNA Sequence Identification Task: It is hard to give a very precise definition of the problem. This can only be explained by giving examples: classifying *bacterial promoters* from *non-promoters*. Given a corpus of DNA sequences, a degree of similarity can be obtained between the test sequence and the corpus by compressing the corpus with the test sequence appended and subtracting the size of this compressed file from the compressed corpus alone	

of times. Once we have all the score cards (corresponding to each scene basis), we can devise clever ways of giving the verdict. Also, note that we can give some indication of confidence of the recognition by refering to the scores obtained.

More sophisticated systems, would further match edges or other attributes of the models with that of the scene to confirm or reject the recognition.

We have logically separated out all the tasks carried out in the geometric hashing process. The actual implementation may combine the various steps or prune them for efficiency and other reasons. Figure 27 shows a block diagram for a possible implementation of the geometric hashing technique.

4. Taxonomy of Computational Biology Problems

In this section we will list a few porblems in Computational Biology and the techniques (which we call tools) currently employed to tackle these problems. The following is only a sample and there are various solutions being offered to some of the problems stated below. It must be emphasized that lot of solutions, which are very promising, are based on statistical methods and is not listed in the following tables. The following is just to indicate the variety in the arsenal that the computer scientists have been employing.

References

1. Booth, K. and Leukar, G. *Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms*, J. Computer and System Sciences, **13**, 335–379.

2. Golumbic Martin Charles, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, 1980.

3. Pavel A. Pevzner, Michael S. Waterman *Open Combinatorial Problems in Computational Molecular Biology*, Proceedings of the Third Israel Symposium on Theory of Computing and Systems, Jan 4-6, 1995, Tel Aviv, Is rael.

4. Richard M. Karp, *Mapping the Genome: Some Combinatorial Problems Arising in Molecular Biology*, 25th ACM STOC '93-5/93/CA, USA.

5. Bonnie Berger, Peter W. Shor, *On the Mathematics of Virus Shell Assembly.*

6. Parsons, D. and Canny, J. *Geometric Problems in Molecular Biology and Robotics*, home page.

7. Jason Tsong-Li Wang, Gung-Wei Chirn, Thomas G. Marr, Bruce Shapiro, Dennis Shasha and Kaizhong Zhang, *Combinatorial Pattern Discovery for Scientific Data: Some Preliminary Results*, home page.

8. Tao Jiang, Eugene L. Lawler, Lusheng Wang, *Aligning Sequences via an Evolutionary Tree: Complexity and Approximation*, STOC 94-5/94 Montreal, Quebec, Canada.

9. Martin Farach, Teresa M. Przytycka, Mikkel Thorup, On the Agreement of Many Trees}, home page.

10. Farach, M. and Thorup, M., *Optimal Evolutionary Tree Comparison by Sparse Dynamic Programming*, 1995, home page.

11. Vineet Bafna and Pavel A. Pevzner *Genome Rearrangements and Sorting by Reversals*, Proc. of the 34th IEEE symposium on the Foundations of Computer Science, 1993.

12. Sridhar Hannenhalli, Pavel Pevzner, *Transforming Cabbage into Turnip*, Proc. of the 27th Annual ACM Symposium on the Theory of Computing 1995.

13. Leonard Adelman, *Molecular Computation of Solutions to Combinatorial Problems*, Science, 1994, **266**, 1021–1024.

14. Richard J. Lipton *Using DNA to Solve NP-Complete Problems.*

15. Richard J. Lipton, Using DNA to Solve SAT.

16. John D. Kececioglu and Eugene W. Myers, *Combinatorial Algorithms for DNA sequence assembly*, TR University of Arizona, 1993, 92–37.

17. Michael S. Waterman, *Introduction to Computational Biology*, Chapman & Hall, 1995.

18. Lamdan, Y. and Wolfson, H. J. Geometric Hashing: A general and efficient model-based rec ognition scheme in *Second IEEE International Conference on Computer Vision*, 1988, 238–249.

19. Levitt, M., Chottia, C., Structural patterns in globular proteins, *Nature*, 1976, **261**, 552–558.

20. Case, D. A., *Computer Simulations of Protein Dynamics and Thermodynamics*, IEEE: Computer Science & Engineering, 1993, 47–57.

21. Philips, A. T., Rosen, J. B. and Walke, V. H., *Molecular Structure Determination by Convex Global Underestimation of Local Energy Minima*, Global Minimization of Nonconvex Energy Functions: Molecular Conformation and Protein Folding, DIMACS Workshop, 1995, 20–21.

22. Searls, D. B., The linguistics of DNA, American Scientist, 1992, **80**(6), 579–591.

23. Idury, R. M. and Waterman, M. S., *A New Algorithm for DNA Sequence Assembly*, J. of Comp. Bio., 1995, **2**(2), pp. 291–306.

24. Lowenstern, D., Hirsh, H., Yianilos, P., and Noordewier, M., *DNA sequence Classification Using Compression-Based Induction*, DIMACS Technical Report, 1995, 95–104.

25. Halperin, D., Kavaraki, L., Latombe, J., Motwani, R., Shelton, C. and Venkatasubramanian, S, *Geometric Manipulation of Flexible Ligands*, Proc. of the 1996 ACM Workshop on Applied Computational Geometry, 1996.

26. Cooper, N. G. (ed), *The Human Genome Project – Deciphering the Blueprint of Heredity*, University Science Books, Mill Valley, California, 1994.

27. Wang, Y., Huff, E., Schwartz, D., *Optical Mapping of site-directed cleavages on single DNA molecules by the RecA-assisted restriction endonuclease technique*, Proc. Nat. Acad. Sci., 1995, **92**, pp 165–169.

28. Meng, X., Benson, K., Chada, K., Huff, E. and Schwartz, D., *Optical mapping of lambda bacteriophage clones using restriction* endonucleases, Nature Genetics, 1995, **9**, pp. 432–438.

29. Lander, E. S. and Waterman, M. S., *Genomic mapping by fingerprinting random clones: a mathematical* analysis, Genomics, 1988, **2**, pp 231–239.

30. Gnedenko, B. V., *The Theory of Probability*, Chelsea Publishers, New York, 1962, pp 122–128.