BIOLOGY COMPUTING

The modern era of molecular biology began with the discovery of the double helical structure of DNA. Today, sequencing nucleic acids, the determination of genetic information at the most fundamental level, is a major tool of biological research (1). This revolution in biology has created a huge amount of data at great speed by directly reading DNA sequences. The growth rate of data volume is exponential. For instance, the volume of DNA and protein sequence data is currently doubling every 22 months (2). One important reason for this exceptional growth rate of biological data is the medical use of such information in the design of diagnostics and therapeutics (3,4). For example, identification of genetic markers in DNA sequences would provide important information regarding which portions of the DNA are significant, and would allow the researchers to find many disease genes of interest (by recognizing them from the pattern of inheritance). Naturally, the large amount of available data poses a serious challenge in storing, retrieving, and analyzing biological information.

A rapidly developing area, *computational biology,* is emerging to meet the rapidly increasing computational need. It consists of many important areas such as information storage, sequence analysis, evolutionary tree construction, protein structure prediction, and so on (3,4). It is playing an important role in some biological research. For example, sequence comparison is one of the most important methodological issues and most active research areas in current *biological sequence analysis.* Without the help of computers, it is almost impossible to compare two or more biological sequences (typically, at least a few hundred characters long).

In this article, we survey recent results on evolutionary tree construction and comparison, computing synthenic distances between multichromosome genomes, and multiple sequence alignment problems.

Evolutionary trees model the evolutionary histories of input data such as a set of species or molecular sequences. Evolutionary trees are useful for a variety of reasons, for example, in homology modeling of (DNA and protein) sequences for diagnostic or therapeutic design, as an aid for devising classifications of organisms, in evaluating alternative hypotheses of adaption, and ancient geographical relationships (5,6). Quite a few methods are known to construct evolutionary trees from the large volume of input data. We will discuss some of these methods in this article. We will also discuss methods for comparing and contrasting evolutionary trees constructed by various methods to find their similarities or dissimilarities, which is of vital importance in computational biology.

Synthenic distance is a measure of distance between multichromosome genomes (where each chromosome is viewed as a set of genes). Applications of computing distances between genomes can be traced back to the well-known *Human Genome Project,* whose objective is to decode this entire DNA sequence and to find the location and ordering of genetic markers along the length of the chromosome. These genetic markers can be used, for example, to trace the inheritance of chromosomes in families and thereby to find the location of disease genes. Genetic markers can be found by finding DNA polymorphisms—that is, locations where two DNA sequences "spell" differently. A key step in finding DNA polymorphisms is the calculation of the *genetic distance,* which is a measure of the correlation (or similarity) between two genomes.

Multiple sequence alignment is an important tool for sequence analysis. It can help extracting and finding biologically important commonalities from a set of sequences. Many versions have been proposed, and a huge number of papers have been written on effective and efficient methods for constructing multiple sequence alignment. We will discuss some of the important versions such as *SP alignment, star alignment, tree alignment, generalized tree alignment,* and *fixed topology alignment with recombination.* Recent results on these versions are given.

We assume that the reader has a basic knowledge of algorithms and computational complexity, such as NP, P, and MAX-SNP. Consult, for example, Refs. 7–9 otherwise.

The rest of this article is organized as follows. In the section entitled "Construction and Comparison of Evolutionary Trees," we discuss construction and comparison methods for evolutionary trees. In the section entitled "Computing Distances Between Genomes," we discuss briefly various distances for comparing sequences and explain in details the synthenic distance measure. In the section entitled "Multiple Sequence Alignment Problems," we discuss multiple sequence alignment problems. We conclude with a few open problems in the section entitled "Conclusion."

CONSTRUCTION AND COMPARISON OF EVOLUTIONARY TREES

The evolution history of organisms is often conveniently represented as trees, called *phylogenetic trees* or simply *phylogenies.* Such a tree has uniquely labeled leaves and unlabeled interior nodes, can be *unrooted* or *rooted* if the evolutionary origin is known, and usually has internal nodes of degree 3.

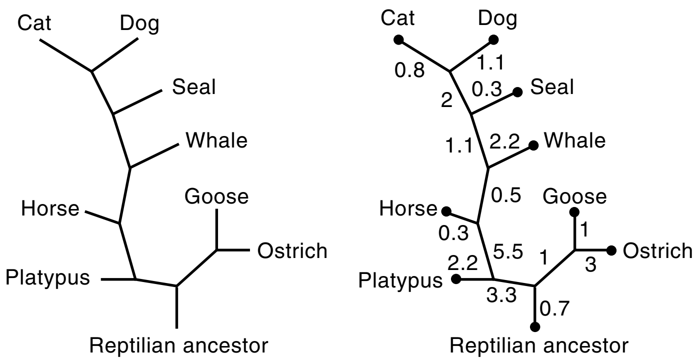

Figure 1. Examples of weighted and unweighted phylogenies.

Figure 1 shows an example of a phylogeny. A phylogeny may also have *weights* on its edges, where an edge weight (more popularly known as *branch length* in genetics) could represent the evolutionary distance along the edge. Many phylogeny reconstruction methods, including the distance and maximum likelihood methods, actually produce weighted phylogenies. Figure 1 also shows a weighted phylogeny (the weights are for illustrative purposes only).

Phylogenetic Construction Methods

Phylogenetic construction methods use the knowledge of evolution of molecules to infer the evolutionary history of the species. The knowledge of evolution is usually in the form of two kinds of data commonly used in phylogeny inference—namely, character matrices where each position (i, j) is base j in sequence i, and distance matrices where each position (i, j) contains the computed distance between sequence i and sequence j. Three major types of phylogenetic construction methods are the *parsimony and compatibility method,* the *distance method,* and the *maximum-likelihood method.* Below we discuss each of them very briefly. See the excellent surveys in Refs. 10 and 11 for more details.

Parsimony methods construct phylogenetic trees for the given sequences such that, in some sense, the total number of changes (i.e., base substitutions) or some weighted sum of the changes is minimized. See Refs. 12–14 for some of the relevant papers.

Distance methods (15–17) try to fit a tree to a matrix of pairwise distances between a set of n species. Entries in the distance matrices are assumed to represent evolutionary distance between species represented by the sequences in the tree, that is, the total number of mutations in both lineages since divergence from the common ancestor. If no tree fits the distance matrix perfectly, then a measure of the discrepancy of the distances in the distance matrix and those in the tree is taken, and the tree with the minimum discrepancy is selected as the best tree. An example of the measure of the discrepancy, which has been used in the literature (15,16), is a weighted least-square measure—that is, of the form

$$\sum_{1 \le i,\, j \le n} w_{ij}(D_{ij} - d_{ij})^2$$

where D_{ij} are the given distances and d_{ij} are the distances computed from the tree.

Maximum-likelihood methods (12,18,19) rely on the statistical method of choosing a tree that maximizes the likelihood—that is, maximizes the probability that the observed data would have occurred. Although this method is quite general and powerful, it is computationally intensive because of the complexity of the likelihood function.

All the above methods have been investigated by simulation and theoretical analysis. None of the methods work well under all evolutionary conditions, but each works well in particular situations. Hence, one must choose the appropriate phylogeny construction method carefully for best results (6).

Comparing Evolutionary Trees

As discussed in the previous section, over the past few decades, many approaches for reconstructing evolutionary trees have been developed, including (not exhaustively) parsimony, compatibility, distance, and maximum-likelihood methods. As a result, in practice they often lead to different trees on the same set of species (20). It is thus of interest to compare evolutionary trees produced by different methods, or by the same method on different data. Several distance models for evolutionary trees have been proposed in the literature. Among them, the best known is perhaps the *nearest-neighbor interchange* (NNI) distance introduced independently in Refs. 21 and 22. Other distances include (a) the *subtree-transfer* distance introduced in Refs. 23 and 24, and (b) the *linear-cost subtree-transfer distance* (25,26). Below, we discuss very briefly a few of these distances.

Nearest-Neighbor Interchange Distance

An NNI operation swaps two subtrees that are separated by an internal edge (u, v), as shown in Fig. 2. The NNI operation is said to *operate* on this internal edge. The NNI distance, $D_{\mathrm{NNI}}(T_1, T_2)$, between two trees T_1 and T_2 is defined as the minimum number of NNI operations required to transform one tree into the other. Culik and Wood (27) [improved later by Li et al. (28)] proved that $n \log n + O(n)$ NNI moves are sufficient to transform a tree of n leaves to any other tree with the same set of leaves. Sleator et al. (29) proved an $\Omega(n \log n)$ lower bound for most pair of trees. Although the distance has been studied extensively in the literature (21,22,27–34), the computational complexity of computing it has puzzled the research community for nearly 25 years until recently when DasGupta et al. (25) showed this problem to be NP-hard. An erroneous proof of the NP-hardness of the NNI distance between unlabeled trees was published in Ref. 34. Since computing the NNI distance is shown to be NP-hard,

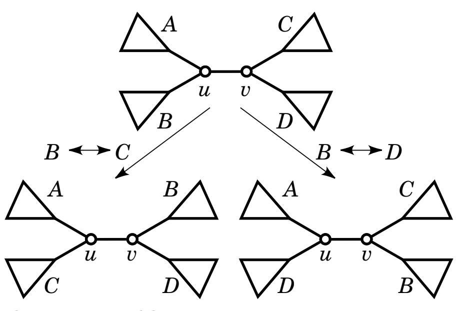

Figure 2. The two possible NNI operations on an internal edge (u, v): exchange $B \leftrightarrow C$ or $B \leftrightarrow D$.

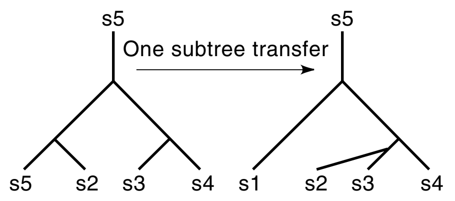

Figure 3. An example of subtree-transfer operation on a tree.

the next obvious question is: *Can we get a good approximation of the distance?* Li et al. (28) show that the NNI distance can be approximated in polynomial time within a factor of log n + $O(1)$.

Subtree-Transfer Distances

An NNI operation can also be viewed as moving a subtree past a neighboring internal node. A more general operation is to transfer a subtree from one place to another arbitrary place. Figure 3 shows such a *subtree-transfer* operation. The subtree-transfer distance, $D_{st}(T_1, T_2)$, between two trees T_1 and T_2 is the minimum number of subtrees we need to move to transform T_1 into T_2 (23–25,35).

It is sometimes appropriate in practice to discriminate among subtree-transfer operations as they occur with different frequencies. In this case, we can charge each subtree-transfer operation a cost equal to the distance (the number of nodes passed) that the subtree has moved in the current tree. The *linear-cost* subtree-transfer distance, $D_{lcst}(T_1, T_2)$, between two trees T_1 and T_2 is then the minimum total cost required to transform T_1 into T_2 by subtree-transfer operations (25,26). Clearly, both subtree-transfer and linear-cost subtree-transfer models can also be used as alternative measures for comparing evolutionary trees generated by different tree reconstruction methods. In fact, on unweighted phylogenies, the linear-cost subtree-transfer distance is identical to the NNI distance (26).

Hein et al. (35) show that computing the subtree-transfer distance between two evolutionary trees is NP-hard and give an approximation algorithm for this distance with performance ratio 3.

Rotation Distance

Rotation distance is a variant of the NNI distance for rooted, ordered trees. A *rotation* is an operation that changes one rooted binary tree into another with the same size. Figure 4 shows the general rotation rule. An easy approximation algorithm for computing distance with a performance ratio of 2 is given in Ref. 36. However, it is not known if computing this distance is NP-hard or not.

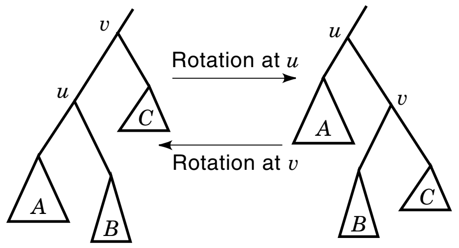

Figure 4. Left and right rotation operations on a rooted binary tree.

Distances on Weighted Phylogenies

Comparison of weighted evolutionary trees has recently been studied in Ref. 20. The distance measure adopted is based on the difference in the partitions of the leaves induced by the edges in both trees, and it has the drawback of being somewhat insensitive to the tree topologies. Both the linear-cost subtree-transfer and NNI models can be naturally extended to weighted trees. The extension for NNI is straightforward: An NNI is simply charged a cost equal to the weight of the edge it operates on. In the case of linear-cost subtree transfer, although the idea is immediate—that is, a moving subtree should be charged for the weighted distance it travels—the formal definition needs some care and can be found in Ref. 26.

Since computing the NNI distance on unweighted phylogenies is NP-hard, it is obvious that computing this distance is NP-hard for weighted phylogenies also. DasGupta et al. (26) give an approximation algorithm for the linear-cost subtree-transfer distance on weighted phylogenies with performance ratio 2. In Ref. 25, the authors give an approximation algorithm for the NNI distance on weighted phylogenies with performance ratio of $O(\log n)$. It is open whether the linear-cost subtree-transfer problem is NP-hard for weighted phylogenies. However, it has been shown that the problem is NP-hard for weighted trees with nonuniquely labeled leaves (26).

COMPUTING DISTANCES BETWEEN GENOMES

The definition and study of appropriate measures of distance between pairs of species is of great importance in computational biology. Such measures of distance can be used, for example, in phylogeny construction, and in taxonomic analysis.

As more and more molecular data become available, methods for defining distances between species have focused on such data. One of the most popular distance measures is the edit distance between homologous DNA or amino acid sequences obtained from different species. Such measures focus on point mutations and define the distance between two sequences as the minimum number of these moves required to transform one sequence into another. It has been recognized that the edit distance may underestimate the distance between two sequences because of the possibility that multiple point mutations occurring at the same locus will be accounted for simply as one mutation. The problem is that the probability of a point mutation is not low enough to rule out this possibility.

Recently, there has been a spate of new definitions of distance that try to treat rarer, macrolevel mutations as the basic moves. For example, if we know the order of genes on a chromosome for two different species, we can define the *reversal* distance between the two species to be the number of reversals of portions of the chromosome to transform the gene order in one species to the gene order in the other species. The question of finding the reversal distance was first explored in the computer science context by Kececioglu and Sankoff and by Bafna and Pevzner, and there has been significant progress made on this question by Bafna, Hannenhalli, Kececioglu, Pevzner, Ravi, Sankoff, and others (37–41). Other moves besides reversals have been considered as well. Breaking off a portion of the chromosome and inserting it elsewhere in the chromosome is referred to as a *transposition,* and one can similarly define the transposition distance (42). Similarly,

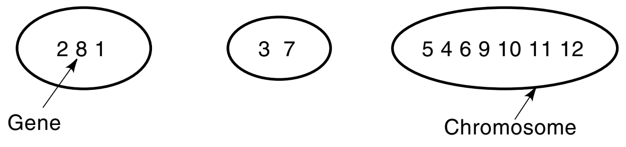

Figure 5. A genome with 12 genes and 3 chromosomes.

allowing two chromosomes (viewed as strings of genes) to exchange suffixes (or sometimes a suffix with a prefix) is known as a *translocation,* and this move can also be used to define an appropriate measure of distance between two species for which much of the genome has been mapped (43).

Ferretti et al. (44) proposed a distance measure that is at an even higher level of abstraction. Here even the order of genes on a particular chromosome of a species is ignored or presumed to be unknown. It is assumed that the genome of a species is given as a collection of sets. Each set in the collection corresponds to a set of genes that are on one chromosome, and different sets in the collection correspond to different chromosomes (see Fig. 5). In this scenario, one can define a move to be either an exchange of genes between two chromosomes, the fission of one chromosome into two, or the fusion of two chromosomes into one (see Fig. 6). The *syntenic distance* between two species has been defined by Ferretti et al. (44) to be the number of such moves required to transform the genome of one species to the genome of the other.

Notice that any recombination of two chromosomes is permissible in this model. By contrast, the set of legal translocations (in the translocation distance model) is severely limited by the order of genes on the chromosomes being translocated. Furthermore, the transformation of the first genome into the second genome does not have to produce a specified order of genes in the second genome. The underlying justification of this model is that the exchange of genes between chromosomes is a much rarer event than the movement of genes within a chromosome and hence a distance function should measure the minimum number of such exchanges needed.

In Ref. 45, the authors prove various results on the syntenic distance. For example, they show that computing the syntenic distance exactly is NP-hard, there is a simple polynomial time approximation algorithm for the synteny problem with performance ratio 2, and computing the syntenic distance is fixed parameter tractable.

The median problem arises in connection with the phylogenetic inference problem (44) and defined as follows. Given three genomes $\mathscr{G}_1$, $\mathscr{G}_2$, and $\mathscr{G}_3$, we are required to construct a genome $\mathscr{G}$ such that the *median distance* $\alpha_{\mathscr{G}} = \sum_{i=1}^{3} D(\mathscr{G}, \mathscr{G}_i)$ is minimized (where D is the syntenic distance). Without any additional constraints, this problem is trivial, since we can take $\mathscr{G}$ to be empty (and then $\alpha_{\mathscr{G}} = 0$). In the context of syntenic distance, any one of the following three constraints seem relevant: (c1) $\mathscr{G}$ must contain all genes present in *all the three* given genomes, (c2) $\mathscr{G}$ must contain all genes present in *at least two* of the three given genomes, (c3) $\mathscr{G}$ must contain all genes present in *at least one* of the three given genomes. Then, computing the median genome is NP-hard with any one of the three constraints (c1), (c2), or (c3). Moreover, one can approximate the median problem in polynomial time [under any one of the constraints (c1), (c2), or (c3)] with a constant performance ratio. See Ref. 45 for details.

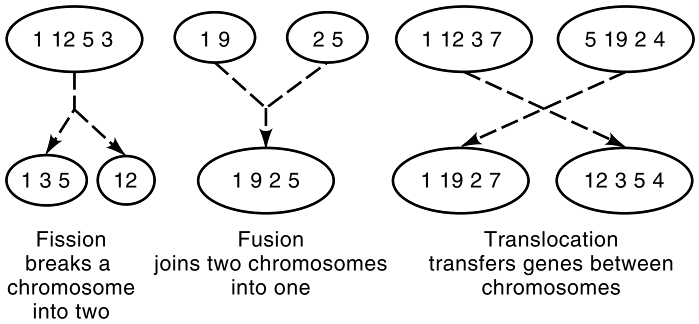
1 12 5 3
1 3 5
12
1 9
2 5
1 9 2 5
1 12 3 7
5 19 2 4
1 19 2 7
12 3 5 4
Fission breaks a chromosome into two
Fusion joins two chromosomes into one
Translocation transfers genes between chromosomes

Figure 6. Different mutation operations.

MULTIPLE SEQUENCE ALIGNMENT PROBLEMS

Multiple sequence alignment is the most critical cutting-edge tool for sequence analysis. It can help extracting, finding, and representing biologically important commonalities from a set of sequences. These commonalities could represent some highly conserved subregions, common functions, or common structures. Multiple sequence alignment is also very useful in inferring the evolutionary history of a family of sequences (46–49).

A *multiple alignment* $\mathscr{A}$ of $k \geq 2$ sequences is obtained as follows: Spaces are inserted into each sequence so that the resulting sequences s_i' ($i = 1, 2, \ldots, k$) have the same length l, and the sequences are arranged in k rows of l columns each.

The value of the multiple alignment $\mathscr{A}$ is defined as

$$\sum_{i=1}^{l} \mu(s_1'(i),\, s_2'(i),\, \ldots s_k'(i))$$

where $s_l'(i)$ denotes the ith letter in the resulting sequence s_l', and $\mu(s_1'(i), s_2'(i), \ldots, s_k'(i))$ denotes the score of the ith column. The multiple sequence alignment problem is to construct a multiple alignment minimizing its value.

Many versions have been proposed based on different objective functions. We will discuss some of the important ones.

SP Alignment and Steiner Consensus String

For *SP score* (sum-of-the-pairs), the score of each column is defined as

$$\mu(s_1'(i),\, s_2'(i),\, \ldots,\, s_k'(i)) = \sum_{1 \leq j < l \leq k} \mu(s_j'(i), s_l'(i))$$

where $\mu(s_j'(i), s_l'(i))$ is the score of the two opposing letters $s_j'(i)$ and $s_l'(i)$. The SP score is sensible and has previously been studied extensively.

The SP-alignment problem is to find an alignment with the smallest SP score. It is first studied in Ref. 52 and subsequently used in Refs. (50,51,53,54). SP alignment problem can be solved exactly by using dynamic programming. However, if there are k sequences and the length of sequences is n, it takes $O(n^k)$ time. Thus, it works for only small numbers of sequences. Some techniques to reduce the time and space have been developed in Refs. 50,55–57. With these techniques, it is possible to optimally align up to six sequences of 200 characters in practice.

In fact, the SP-alignment problem is NP-hard (58). Thus, it is impossible to have a polynomial time algorithm for this problem. In the proof of NP-hardness, it is assumed that some pairs of identical characters have nonzero score. An interesting open problem is, What if each pair of two identical characters is scored 0?

The first approximation algorithm was given by Gusfield (53). He introduced the *center star* algorithm. Center star algorithm is very simple and efficient. It selects a sequence (called *center string*) s_c in the set of k given sequences S such that $\sum_{i=1}^{k} dist(s_c, s_i)$ is minimized. It then optimally aligns the sequences in $S - \{s_c\}$ to s_c and gets $k - 1$ pairwise alignments. These $k - 1$ pairwise alignments lead to a multiple alignment for the k sequences in S. If the score scheme for pairs of characters satisfies the triangle inequality, the cost of the multiple alignment produced by the center star algorithm is at most twice of the optimum (47,53). Some improved results were reported in Refs. 54 and 59.

Another score called *consensus* score is defined as follows:

$$\mu(s'_1(i),\, s'_2(i),\, \ldots s'_k(i)) = \min_{s\in\Sigma} \sum_{j=1}^{k} \mu(s'_j(i),\, s)$$

where Σ is the set of characters that form the sequences. Here we reconstruct a character for each column and thus obtain a string. This string is called a *Steiner consensus string* and can be used as a representative for the set of given sequences. The problem is called the *Steiner consensus string* problem.

The Steiner consensus string problem was proved to be NP-complete (60) and MAX SNP-hard (58). In the proof of MAX SNP-hardness, it is assumed that there is a "wild card," and thus the triangle inequality does not hold. Combining with the results in Ref. 61, it shows that there is no polynomial time approximation scheme for this problem. Interestingly, the same center star algorithm also has performance ratio 2 for this problem (47).

Tree Alignment

Tree Score. In order to define the score $\mu(s'_1(i), s'_2(i), \ldots s'_k(i))$ of the ith column, an *evolutionary* (or *phylogenetic*) tree $T = (V, E)$ with k leaves is assumed, with each leaf j corresponding to a sequence s_j. (Here V and E denote the sets of nodes and edges in T, respectively.) Let $k + 1$, $k + 2$, . . ., $k + m$ be the internal nodes of T. For each internal node j, reconstruct a letter (possibly a space) $s'_j(i)$ such that $\sum_{(p,q)\in E}\mu(s'_p(i), s'_q(i))$ is minimized. The score $\mu(s'_1(i), s'_2(i), \ldots, s'_k(i))$ of the ith column is thus defined as

$$\mu(s'_1(i),\, s'_2(i),\, \ldots s'_k(i)) = \sum_{(p,\,q)\in E} \mu(s'_p(i),\, s'_q(i))$$

This measure has been discussed in Refs. 14, 48, 50, 59, and 62. Multiple sequence alignment with tree score is often referred to as *tree alignment* in the literature.

Note that a tree alignment induces a set of *reconstructed* sequences, each corresponding to an internal node. Thus, it is convenient to reformulate tree alignment as follows: Given a set X of k sequences and an evolutionary tree T with k leaves, where each leaf is associated with a given sequence, reconstruct a sequence for each internal node to minimize the *cost* of T. Here, the cost of T is the sum of the edit distance of each pair of (given or reconstructed) sequences associated with an edge. Observe that, once a sequence for each internal node has been reconstructed, a multiple alignment can be obtained by optimally aligning the pair of sequences associated with each edge of the tree. Moreover, the tree score of this induced multiple alignment equals the cost of T. In this sense, the two formulations of tree alignment are equivalent.

Sankoff gave an exact algorithm for tree alignment that runs in $O(n^k)$, where n is the length of the sequences and k is the number of given sequences. Tree alignment was proved to be NP-hard (58).

Therefore it is unlikely to have a polynomial time algorithm for tree alignment. Some heuristic algorithms have also been considered in the past. Altschul and Lipman (50) tried to cut down the computation volume required by dynamic programming. Sankoff, Cedergren, and Lapalme gave an iterative improvement method to speed up the computation (48,62). Waterman and Perlwitz devised a heuristic method when the sequences are related by a binary tree (64). Hein (65,66) proposed a heuristic method based on the concept of a *sequence graph*. Ravi and Kececioglu (67) designed an approximation algorithm with performance ratio $(deg + 1)/(deg - 1)$ when the given tree is a *regular deg-ary* tree (i.e., each internal node has exactly *deg* children).

The first approximation algorithm with a guaranteed performance ratio was devised by Wang, Jiang, and Lawler (63). A ratio 2 algorithm was given. The algorithm was then extended to a polynomial time approximation scheme (PTAS); that is, the performance ratio could arbitrarily approach 1. The PTAS requires computing exact solutions for depth-t subtrees. For a fixed t, the performance ratio was proved to be $1 + 3/t$, and the running time was proved to be $O((k/deg^t)^{deg^{t-1}+2}M(2, t - 1, n))$, where deg is the degree of the given tree, and $M(deg, t - 1, n)$ is the time needed to optimally align a tree with $deg^{t-1} + 1$ leaves, which is upper-bounded by $O(n^{deg^{t-1}+1})$. Based on the analysis, to obtain a performance ratio less than 2, exact solutions for depth-4 subtrees must be computed, and thus optimally aligning nine sequences at a time is required. This is impractical even for sequences of length 100.

An improved version was given in Ref. 68. They proposed a new PTAS for the case where the given tree is a regular *deg*-ary tree. The algorithm is much faster than the one in Ref. 63. The algorithm also must do local optimizations for depth-t subtrees. For a fixed t, the performance ratio of the new PTAS is $1 + 2/t - 2/t2^t$ and the running time is $O(\min\{2^t, k\}kdM(deg, t - 1, n))$, where d is the depth of the tree. Presently, there are efficient programs (62) to do local optimizations for three sequences ($t = 2$). In fact, we can expect to obtain optimal solutions for five sequences ($t = 3$) of length 200 in practice since there is such a program (55,56) for SP score, and similar techniques can be used to attack tree alignment problem. Therefore, solutions with costs at most 1.583 times the optimum can be obtained in practice for strings of length 200.

For tree alignment, the given tree is typically a binary tree. Recently, Wang, Jiang, and Gusfield designed a PTAS for binary trees. The new approximation scheme adopts a more clever partitioning strategy and has a better time efficiency for the same performance ratio. For any fixed r, where $r = 2^{t-1} + 1 - q$ and $0 \leq q \leq 2^{t-2} - 1$, the new PTAS runs in time $O(kdn^r)$ and achieves an approximation ratio of $2^{t-1}/[2^{t-2}(t + 1) - q]$. Here the parameter r represents the "size" of local optimization. In particular, when $r = 2^{t-1} + 1$, its approximation ratio is simply $2/(t + 1)$.

Generalized Tree Alignment

In practice, we often face a more difficult problem called *generalized tree alignment.* Suppose we are given a set of sequences. The problem is to construct an evolutionary tree as well as a set of sequences (called reconstructed sequences) such that each leaf of the evolutionary tree is assigned a given sequence, each internal node of the tree is assigned a reconstructed sequence, and the cost of the tree is minimized over all possible evolutionary trees and reconstructed sequences.

Intuitively, the problem is harder than tree alignment since the tree is not given and we have to compute the tree structure as well as the sequences assigned to internal nodes. In fact, the problem was proved to be MAX SNP-hard (58) and a simplified proof was given in Ref. 69. It implies that it is impossible to have a PTAS for generalized tree alignment unless P = NP (61). This confirms the observation from approximation point of view.

Generalized tree alignment problem is in fact the Steiner tree problem in sequence spaces. One might use the approximation algorithms with guaranteed performance ratios (70) for graph Steiner trees. However, this may lead to a tree structure where a given sequence is an internal node. Thus, it is impossible to interpret the tree as a phylogeny. Schwikowski and Vingron (71) give a method that combines clustering algorithms and Hein's sequence graph method. The produced solutions contain biologically reasonable trees and keep the guaranteed performance ratio.

Fixed Topology History/Alignment with Recombination

Multigene families, viruses, and alleles from within populations experience recombinations (23,24,72,73). When recombination happens, the ancestral material on the present sequence s_1 is located on two sequences s_2 and s_3. s_2 and s_3 can be cut at k locations (break points) into $k + 1$ pieces, where $s_2 = s_{2,1}s_{2,2} \ldots s_{2,l+1}$ and $s_3 = s_{3,1}s_{3,2} \ldots s_{3,l+1}$. s_1 can be represented as $s_{\hat{2},1}s_{\hat{3},2}s_{\hat{2},3} \ldots s_{\hat{2},i}s_{\hat{3}+1} \ldots$, where subsequences $s_{\hat{2},i}$ and $s_{\hat{3}+1}$ differ from the corresponding $s_{2,i}$ and $s_{3,i+1}$ by insertion, deletion, and substitution of letters. k, the number of times s_1 switches between s_2 and s_3, is called the number of *crossovers.* The cost of the recombination is

$$dist(s_{1,1}, s_{\hat{1},1}) + dist(s_{2,2}, s_{\hat{2},2}), \ldots dist(s_{1,i}, s_{\hat{1},i}) \\ + dist(s_{2,i+1}, s_{2,\hat{i+1}}) + \cdots + k\chi$$

where $dist(s_{2,i+1}, s_{\hat{2}+1})$ is the edit distance between the two sequences $s_{2,i+1}$ and $s_{\hat{2}+1}$, k is the number of crossovers and χ is the crossover penalty. The *recombination* distance to produce s_1 from s_2 and s_3 is the cost of a recombination that has the smallest cost among all possible recombinations. We use $r\_dist(s_1, s_2, s_3)$ to denote the recombination distance. For more details, see Refs. 72 and 74.

When recombination occurs, the given topology is no longer a binary tree. Instead, some nodes, called *recombination nodes,* in the given topology may have two parents (23,24). In a more general case as described in Ref. 72, the topology may have more than one root. The set of roots is called a *protoset.* The edges incident to recombination nodes are called *recombination* edges. See Fig. 7(b). A node/edge is *normal* if it is not a recombination node/edge.

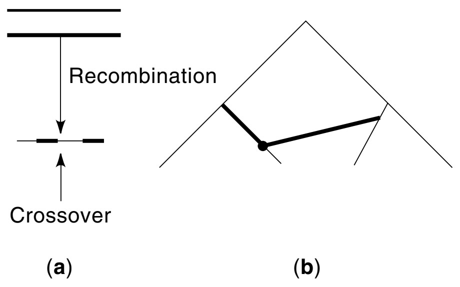

Figure 7. (a) Recombination operation. (b) The topology. The dark edges are recombination edges. The circled node is a recombination node.

The cost of a pair of recombination edges is the recombination distance to produce the sequence on the recombination node from the two sequences on its parents. The cost of other normal edges is the edit distance between two sequences. A topology is *fully labeled* if every node in the topology is labeled. For a fully labeled topology, the cost of the topology is the total cost of edges in the topology. Each node in the topology with degree greater than 1 is an internal node. Each leaf/terminal (degree 1 node) in the topology is labeled with a given sequence. The goal here is to construct a sequence for each internal node such that the cost of the topology is minimized. We call this problem *fixed topology history with recombination* (FTHB).

Obviously, this problem is a generalization of tree alignment. The difference is that the given topology is no longer a binary tree. Instead, there are some recombination nodes which have two parents instead of one. Moreover, there may be more than one root in the topology.

A different version called fixed topology alignment with recombination (FTAR) is also discussed (75). From an approximation point of view, FTHR and FTAR are much harder than tree alignment. It is shown that FTHR and FTAR cannot be approximated within any constant performance ratio unless $P = NP$ (75).

A more restricted case, where each internal node has at most one recombination child and there are at most six parents of recombination nodes in any path from the root to a leaf in the given topology, is also considered. It is shown that the restricted version for both FTHR and FTAR is MAX-SNP-hard. That is, there is no polynomial time approximation scheme unless $P = NP$ (75).

The above hardness results are disappointing. However, recombination occurs infrequently. So, it is interesting to study some restricted cases. A *merge node* of recombination node v is the lowest common ancestor of v's two parents. The two different paths from a recombination node to its merge node are called *merge paths.* We then study the case where

(C1) each internal node has at most one recombination child and

(C2) any two merge paths for different recombination nodes do not share any common node.

Using a method similar to the lifting method for tree alignment, one can get a ratio-3 approximation algorithm for both FTHR and HTAR when the given topology satisfies (C1) and (C2). The ratio-3 algorithm can be extended to a PTAS for FTAR with bounded number of crossovers. (See Ref. 75.)

Remarks. Hein may have been the first to study the method to reconstruct the history of sequences subject to recombination (23,24). Hein observed that the evolution of a sequence with k recombinations could be described by k recombination points and $k + 1$ trees describing the evolution of the $k + 1$ intervals, where two neighboring trees were either identical or differed by one subtree transfer operation (23–26,35). A heuristic method was proposed to find the most parsimonious history of the sequences in terms of mutation and recombination operations.

Another strike was given by Kececioglu and Gusfield (72). They introduced two new problems, namely, *recombination distance* and *bottleneck recombination history*. They tried to include higher-order evolutionary events such as block insertions and deletions (76) and tandem repeats (77,78).

CONCLUSION

In this article we have discussed some important topics in the field of computational biology such as the phylogenetic construction and comparsion methods, synthenic distance between genomes, and the multiple sequence alignment problems. Given the vast majority of topics in computational biology, these discussed topics constitute only a part of them. Some of the important topics which were *not* covered in this chapter are:

- Protein structure prediction
- DNA physical mapping problems
- Metabolic modeling
- String/database search problems, etc.

We hope that this survey article will inspire the readers for further study and research of these and other related topics.

Papers on computational molecular biology have started to appear in many different books, journals, and conferences. Below we list some sources which could serve as excellent starting points for various problems that arise in computational biology:

Books: References 49, 53, 79–83.

Journals: *Computer Applications in the Biosciences (*recently renamed as *Bioinformatics), Journal of Computational Biology, Bulletin of Mathematical Biology, Journal of Theoretical Biology*.

Conferences: *Annual Symposium on Combinatorial Pattern Matching (CPM), Pacific Symposium on Biocomputing (PSB), Annual International Conference on Computational Molecular Biology (RECOMB), Annual Conference on Intelligent Systems in Molecular Biology (ISMB)*.

ACKNOWLEDGMENTS

We thank Prof. Tao Jiang for bringing the authors together. Thanks also go to Dr. Todd Wareham, who carefully read the draft and gave valuable suggestions. The work of Lusheng Wang was supported in part by Hong Kong Research Council. The work of Bhaskar DasGupta was supported in part by a grant from Rutgers Research Council.

BIBLIOGRAPHY

1. M. S. Waterman, Sequence alignments, in M. S. Waterman (ed.), *Mathematical Methods for DNA Sequences,* Boca Raton, FL: CRC Press, 1989, pp. 53–92.
2. W. Miller, S. Scbwartz, and R. C. Hardison, A point of contact between computer science and molecular biology, *IEEE Computat. Sci. Eng.,* **1** (1): 69–78, 1994.
3. K. A. Frenkel, The human genome project and informatics, *Commun. ACM,* **34** (11): 41–51, 1991.
4. E. S. Lander, R. Langridge, and D. M. Saccocio, Mapping and interpreting biological information, *Commun. ACM,* **34** (11): 33–39, 1991.
5. V. A. Funk and D. R. Brooks, *Phylogenetic Systematics as the Basis of Comparative Biology,* Washington, DC: Smithsonian Institution Press, 1990.
6. D. M. Hillis, B. K. Mable, and C. Moritz, Applications of Molecular Systematics, in D. M. Hillis et al. (eds.), *Molecular Systematics,* 2nd ed., Sunderland, MA: Sinauer Associates, 1996, pp. 515–543.
7. M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness,* San Francisco: Freeman, 1979.
8. D. Hochbaum, *Approximation Algorithms for NP-Hard Problems,* PWS Publishers, 1996.
9. C. H. Papadimitriou, *Computational Complexity,* Reading, MA: Addison-Wesley, 1994.
10. J. Felsenstein, Phylogenies from molecular sequences: Inferences and reliability, *Annu. Rev. Genet.,* **22**: 521–565, 1988.
11. D. L. Swofford et al., Phylogenetic Inference, in D. M. Hillis et al., (eds.), *Molecular Systematics,* 2nd ed., Sunderland, MA: Sinauer Associates, 1996, pp. 407–514.
12. A. W. F. Edwards and L. L. Cavalli-Sforza, The reconstruction of evolution, *Ann. Hum. Genet.,* **27**: 105, 1964 (also in *Heredity* **18**: 553, 1964.
13. W. M. Fitch, Toward defining the course of evolution: Minimum change for a specified tree topology, *Syst. Zool.,* **20**: 406–416, 1971.
14. D. Sankoff, Minimal mutation trees of sequences, *SIAM J. Appl. Math.,* **28**: 35–42, 1975.
15. L. L. Cavalli-Sforza and A. W. F. Edwards, Phylogenetic analysis: Models and estimation procedures, *Evolution,* **32**: 550–570, 1967; also published in *Am. J. Hum. Genet.,* **19**: 233–257, 1967.
16. W. M. Fitch and E. Margoliash, Construction of phylogenetic trees, *Science,* **155**: 279–284, 1967.
17. N. Saitou and M. Nei, The neighbor-joining method: A new method for reconstructing phylogenetic trees, *Mol. Biol. Evol.,* **4**: 406–425, 1987.
18. J. Felsenstein, Evolutionary trees for DNA sequences: A maximum likelihood approach, *J. Mol. Evol.,* **17**: 368–376, 1981.
19. D. Barry and J. A. Hartigan, Statistical analysis of hominoid molecular evolution, *Stat. Sci.,* **2**: 191–210, 1987.
20. M. Kuhner and J. Felsenstein, A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates, *Mol. Biol. Evol.,* **11** (3): 459–468, 1994.
21. D. F. Robinson, Comparison of labeled trees with valency three, *J. Comb. Theory Ser. B,* **11**: 105–119, 1971.

22. G. W. Moore, M. Goodman, and J. Barnabas, An iterative approach from the standpoint of the additive hypothesis to the dendrogram problem posed by molecular data sets, *J. Theor. Biol.,* **38**: 423–457, 1973.

23. J. Hein, Reconstructing evolution of sequences subject to recombination using parsimony, *Math. Biosci.,* **98**: 185–200, 1990.

24. J. Hein, A heuristic method to reconstruct the history of sequences subject to recombination, *J. Mol. Evol.,* **36**: 396–405, 1993.

25. B. DasGupta et al., On distances between phylogenetic trees, *Proc. 8th Annu. ACM-SIAM Symp. Discrete Algorithms,* 1997, pp. 427–436.

26. B. DasGupta et al., On the linear-cost subtree-transfer distance, *Algorithmica,* special issue computational biology, 1998, in press.

27. K. Culik II and D. Wood, A note on some tree similarity measures, *Inf. Process. Lett.,* **15**: 39–42, 1982.

28. M. Li, J. Tromp, and L. X. Zhang, On the nearest neighbor interchange distance between evolutionary trees, *J. Theor. Biol.,* **182**: 463–467, 1996.

29. D. Sleator, R. Tarjan, and W. Thurston, Short encodings of evolving structures, *SIAM J. Discrete Math.,* **5**: 428–450, 1992.

30. M. S. Waterman and T. F. Smith, On the similarity of dendrograms, *J. Theor. Biol.,* **73**: 789–800, 1978.

31. W. H. E. Day, Properties of the nearest neighbor interchange metric for trees of small size, *J. Theor. Biol.,* **101**: 275–288, 1983.

32. J. P. Jarvis, J. K. Luedeman, and D. R. Shier, Counterexamples in measuring the distance between binary trees, *Math. Soc. Sci.,* **4**: 271–274, 1983.

33. J. P. Jarvis, J. K. Luedeman, and D. R. Shier, Comments on computing the similarity of binary trees, *J. Theor. Biol.,* **100**: 427–433, 1983.

34. M. Křvánek, Computing the nearest neighbor interchange metric for unlabeled binary trees is NP-complete, *J. Classif.,* **3**: 55–60, 1986.

35. J. Hein et al., On the complexity of comparing evolutionary trees, *Discrete Appl. Math.,* **71**: 153–169, 1996.

36. D. Sleator, R. Tarjan, and W. Thurston, Rotation distance, triangulations, and hyperbolic geometry, *J. Amer. Math. Soc.,* **1**: 647–681, 1988.

37. V. Bafna and P. Pevzner, Genome rearrangements and sorting by reversals, *34th IEEE Symp. Found. Comput. Sci.,* 1993, pp. 148–157.

38. V. Bafna and P. Pevzner, Sorting by reversals: Genome rearrangements in plant organelles and evolutionary history of X chromosome, *Mol. Biol. Evol.,* **12**: 239–246, 1995.

39. S. Hannenhalli and P. Pevzner, Transforming cabbage into turnip (polynomial algorithm for sorting signed permutations by reversals), *Proc. 27th Annu. ACM Symp. Theory Comput.,* 1995, pp. 178–189.

40. J. Kececioglu and D. Sankoff, Exact and Approximation Algorithms for the Inversion Distance between Two Permutations, *Proc. 4th Annu. Symp. Combinatorial Pattern Matching,* Lecture Notes in Computer Science 684, Berlin: Springer-Verlag, 1993, pp. 87–105.

41. J. Kececioglu and D. Sankoff, Efficient Bounds for Oriented Chromosome Inversion Distance, *Proc. 5th Annu. Symp. on Combinatorial Pattern Matching,* Lecture Notes in Computer Science 807, Berlin: Springer-Verlag, 1994, pp. 307–325.

42. V. Bafna and P. Pevzner, Sorting by transpositions, *Proc. 6th Annu. ACM-SIAM Symp. Discrete Algorithms,* 1995, pp. 614–623.

43. J. Kececioglu and R. Ravi, Of mice and men: Evolutionary distances between genomes under translocation, *Proc. 6th Annu. ACM-SIAM Symp. Discrete Algorithms,* 1995, 604–613.

44. V. Ferretti, J. H. Nadeau, and D. Sankoff, Original synteny, *Proc. 7th Annu. Symp. Comb. Pattern Matching,* 1996, pp. 159–167.

45. B. DasGupta et al., On the complexity and approximation of syntenic distance, *1st Annu. Int. Conf. Comput. Mol. Biol.,* 1997, pp. 99–108.

46. S. C. Chan, A. K. C. Wong, and D. K. T. Chiu, A survey of multiple sequence comparison methods, *Bull. Math. Biol.,* **54** (4): 563–598, 1992.

47. D. Gusfield, *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology,* Cambridge: Cambridge Univ. Press, 1997.

48. D. Sankoff and R. Cedergren, Simultaneous Comparisons of Three or More Sequences Related by a Tree, in D. Sankoff and J. Kruskal (eds.), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison,* Reading, MA: Addison-Wesley, 1983, pp. 253–264.

49. M. S. Waterman, *Introduction to Computational Biology: Maps, Sequences, and Genomes,* London: Chapman & Hall, 1995.

50. S. Altschul and D. Lipman, Trees, stars, and multiple sequence alignment, *SIAM J. Appl. Math.,* **49**: 197–209, 1989.

51. D. Baconn and W. Anderson, Multiple sequence alignment, *J. Mol. Biol.,* **191**: 153–161, 1986.

52. H. Carrillo and D. Lipman, The multiple sequence alignment problem in biology, *SIAM J. Appl. Math.,* **48**: 1073–1082, 1988.

53. D. Gusfield, Efficient methods for multiple sequence alignment with guaranteed error bounds, *Bull. Math. Biol.,* **55**: 141–154, 1993.

54. P. Pevzner, Multiple alignment, communication cost, and graph matching, *SIAM J. Appl. Math.,* **56** (6): 1763–1779, 1992.

55. S. Gupta, J. Kececioglu, and A. Schaffer, Making the shortest-paths approach to sum-of-pairs multiple sequence alignment more space efficient in practice, *Proc. 6th Symp. Comb. Pattern Matching, Springer LNCS937,* 1995, 128–143.

56. J. Lipman, S. F. Altschul, and J. D. Kececioglu, A tool for multiple sequence alignment, *Proc. Natl. Acad. Sci. USA,* **86**: 4412–4415, 1989.

57. G. D. Schuler, S. F. Altschul, and D. J. Lipman, A workbench for multiple alignment construction and analysis, in *Proteins: Structure, Function and Genetics,* in press.

58. L. Wang and T. Jiang, On the complexity of multiple sequence alignment, *J. Comput. Biol.,* **1**: 337–348, 1994.

59. V. Bafna, E. Lawer, and P. Pevzner, Approximate methods for multiple sequence alignment, *Proc. 5th Symp. Comb. Pattern Matching, Springer LNCS 807,* 1994, pp. 43–53.

60. E. Sweedyk and T. Warnow, The tree alignment problem is NP-complete, unpublished manuscript.

61. S. Arora et al., On the intractability of approximation problems, *33rd IEEE Symp. Found. Comput. Sci.,* 1992, pp. 14–23.

62. D. Sankoff, R. J. Cedergren, and G. Lapalme, Frequency of insertion, deletion, transversion, and transition in the evolution of 5S ribosomal RNA, *J. Mol. Evol.,* **7**: 133–149, 1976.

63. L. Wang, T. Jiang, and E. L. Lawler, Approximation algorithms for tree alignment with a given phylogeny, *Algorithmica,* **16**: 302–315, 1996.

64. M. S. Waterman and M. D. Perlwitz, Line geometries for sequence comparisons, *Bull. Math. Biol.,* **46**: 567–577, 1984.

65. J. Hein, A tree reconstruction method that is economical in the number of pairwise comparisons used, *Mol. Biol. Evol.,* **6** (6): 669–684, 1989.

66. J. Hein, A new method that simultaneously aligns and reconstructs ancestral sequences for any number of homologous sequences, when the phylogeny is given, *Mol. Biol. Evol.,* **6**: 649–668, 1989.

67. R. Ravi and J. Kececioglu, Approximation algorithms for multiple sequence alignment under a fixed evolutionary tree, *5th Annu. Symp. Comb. Pattern Matching,* 1995, pp. 330–339.

68. L. Wang and D. Gusfield, Improved approximation algorithms for tree alignment, *J. Algorithms,* **25**: 255–173, 1997.

69. H. T. Wareham, A simplified proof of the NP-hardness and MAX SNP-hardness of multiple sequence tree alignment, *J. Computat. Biol.,* **2**: 509–514, 1995.

70. A. Z. Zelikovsky, The 11/6 approximation algorithm for the Steiner problem on networks, *Algorithmica,* **9**: 463–470, 1993.

71. B. Schwikowski and M. Vingron, The deferred path heuristic for the generalized tree alignment problem, *1st Annu. Int. Conf. Comput. Mol. Biol.,* 1997, pp. 257–266.

72. J. Kececioglu and D. Gusfield, Reconstructing a history of recombinations from a set of sequences, *5th Annu. ACM-SIAM Symp. Discrete Algorithms,* 1994, pp. 471–480.

73. F. W. Stahl, Genetic recombination, *Sci. Amer.,* **256** (2): 90–101, 1987.

74. J. D. Watson et al., *Molecular Biology of the Gene,* 4th ed., Menlo Park, CA: Benjamin-Cummings, 1987.

75. B. Ma, L. Wang, and M. Li, Fixed topology alignment with recombination, submitted.

76. Z. Galil and R. Ciancarlo, Speeding up dynamic programming with applications to molecular biology, *Theor. Comput. Sci.,* **64**: 107–118, 1989.

77. S. Kannan and E. W. Myers, An algorithm for locating non-overlapping regions of maximum alignment score, *3rd Annu. Symp. Comb. Pattern Matching,* 1993, pp. 74–86.

78. G. M. Landau and J. P. Schmidt, An algorithm for approximate tandem repeats, *3rd Annu. Symp. Comb. Pattern Matching,* 1993, pp. 120–133.

79. J. Collado-Vides, B. Magasanik, and T. F. Smith (eds.), *Integrative Approaches to Molecular Biology,* Cambridge, MA: MIT Press, 1996.

80. L. Hunter (ed.), *Artificial Intelligence in Molecular Biology,* Cambridge, MA: MIT Press, 1993.

81. J. Meidanis and J. C. Setubal, *Introduction to Computational Molecular Biology,* Boston: PWS Publishing, 1997.

82. D. Sankoff and J. Kruskal (eds.), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison,* Reading, MA: Addison-Wesley, 1983.

83. G. A. Stephens, *String Searching Algorithms,* Singapore: World Scientific, 1994.

BHASKAR DASGUPTA
Rutgers University
LUSHENG WANG
City University of Hong Kong