Chapter 1
Introduction

1.1 Biological Motivations

A vast diversity of organisms exist on Earth, but they are amazingly similar. Every organism is made up of cells and depends on two types of molecules: DNA and proteins. DNA in a cell transmits itself from generation to generation and holds genetic instructions that guide the synthesis of proteins and other molecules. Proteins perform biochemical reactions, pass signals among cells, and form the body's components. DNA is a sequence of 4 nucleotides, whereas proteins are made of 20 amino acids arranged in a linear chain.

Due to the linear structure of DNA and protein, sequence comparison has been one of the common practices in molecular biology since the first protein sequence was read in the 1950s. There are dozens of reasons for this. Among these reasons are the following: sequence comparisons allow identification of genes and other conserved sequence patterns; they can be used to establish functional, structural, and evolutionary relationship among proteins; and they provide a reliable method for inferring the biological functions of newly sequenced genes.

A striking fact revealed by the abundance of biological sequence data is that a large proportion of genes in an organism have significant similarity with ones in other organisms that diverged hundreds of millions of years ago. Two mammals have as many as 99% genes in common. Humans and fruit flies even have at least 50% genes in common.

On the other hand, different organisms do differ to some degree. This strongly suggests that the genomes of existing species are shaped by evolution. Accordingly, comparing related genomes provides the best hope for understanding the language of DNA and for unraveling the evolutionary relationship among different species.

Two sequences from different genomes are homologous if they evolved from a common ancestor. We are interested in homologies because they usually have conserved structures and functions. Because species diverged at different time points, homologous proteins are expected to be more similar in closely related organisms than in remotely related ones.

When a new protein is found, scientists usually have little idea about its function. Direct experimentation on a protein of interest is often costly and time-consuming. As a result, one common approach to inferring the protein's function is to find by similarity search its homologies that have been studied and are stored in database. One remarkable finding made through sequence comparison is about how cancer is caused. In 1983, a paper appearing in *Science* reported a 28-amino-acid sequence for platelet derived growth factors, a normal protein whose function is to stimulate cell growth. By searching against a small protein database created by himself, Doolittle found that the sequence is almost identical to a sequence of υ-sis, an oncogene causing cancer in woolly monkeys. This finding changed the way oncogenesis had been seen and understood. Today, it is generally accepted that cancer might be caused by a normal growth gene if it is switched on at the wrong time.

Today, powerful sequence comparison methods, together with comprehensive biological databases, have changed the practice of molecular biology and genomics. In the words of Gilbert, Nobel prize winner and co-inventor of practical DNA sequencing technology,

> The new paradigm now emerging, is that all the 'genes' will be known (in the sense of being resident in databases available electronically), and that the starting point of biological investigation will be theoretical. (Gilbert, 1991, [76])

1.2 Alignment: A Model for Sequence Comparison

1.2.1 Definition

Alignment is a model used by biologists for bringing up sequence similarity. Let Σ be an alphabet, and let $X = x_1x_2\ldots x_m$ and $Y = y_1y_2\ldots y_n$ be two sequences over Σ. We extend Σ by adding a special symbol '-', which denotes *space*. An *alignment* of sequences X and Y is a two-row matrix A with entries in $\Sigma \cup \{-\}$ such that

1. The first (second) row contains the letters of X (Y) in order;
2. One or more spaces '-' may appear between two consecutive letters of Σ in each row;
3. Each column contains at least one letter of Σ.

For example, an alignment between a fragment of the yeast glutathione S-transferase I (GST1) protein sequence and the viral-enhancing factor (VEF) protein sequence is shown in Figure 1.1.

In an alignment, columns that contain two letters are called *matches* if the letters are identical and *mismatches* otherwise. Columns containing a space are called *indels*. In order to be accurate, we sometimes call the columns *insertions* if they contain a space in the first row and *deletions* if they contain a space in the second row. In the alignment of sequences GST1 and VEF given in Figure 1.1, there are 37 matches highlighted by using vertical bars in the central line. Besides, there are 3 insertions and 15 deletions.

```
GST1 SRAFRLLWLLDHLNLEYEIVPYKRDANFRAPPELKKIHPLGRSPLLEVQDRETGKKKILA
     |  ||   | |   ||  |||    |  |        |       | |  | |
VEF  SYLFRFRGLGDFMLLELQIVPILNLASVRVGNHHNGPHSYFNTTYLSVEVRDT-------

GST1 ESGFIFQYVL---QHFDHSHVLMSEDADIADQINYYLFYVEGSLQPPLMIEFILSKVKDS
       |  | |         | |                ||  |    |      |  |  |
VEF  SGGVVFSYSRLGNEPMTHEH----HKFEVFKDYTIHLFIQE----PGQRLQLIVNKTLDT

GST1 GMPFPISYLARKVADKISQAYSSGEVKNQFDFV
       |      ||  |        |       |||
VEF  ALPNSQNIYARLTATQLVVGEQSIIISDDNDFV
```

Fig. 1.1 An example of protein sequence alignment.

From a biological point of view, an alignment of two sequences poses a hypothesis on how the sequences evolved from their closest common ancestor. An alignment accounts for the following three mutational events:

- Substitution (also called point mutation) - A nucleotide is replaced by another.
- Insertion - One or several nucleotides are inserted at a position in a sequence.
- Deletion - One of several nucleotides are deleted from a sequence.

In an alignment, a mismatch represents a hypothetical substitution event, and an indel $\binom{-}{a}$ or $\binom{a}{-}$ represents that a is added or deleted in an insertion or deletion event.

We often say an alignment is *gapped* or *ungapped* to emphasize whether indels are allowed or not in the alignment.

1.2.2 Alignment Graph

There are many ways of aligning two sequences. By using theoretic-graph concepts, however, we can obtain a compact representation of these alignments.

Consider two sequences of length m and n. The *alignment graph* of these two sequences is a direct graph G. The vertices of G are lattice points (i, j), $0 \leq i \leq m$ and $0 \leq j \leq n$ in the $(m+1) \times (n+1)$ grid, and there is an edge from vertex (i, j) to another (i', j') if and only if $0 \leq i' - i \leq 1$ and $0 \leq j' - j \leq 1$. Figure 1.2 gives the alignment graph for sequences `ATACTGG` and `GTCCGTG`. As shown in Figure 1.2, there are vertical, horizontal, and diagonal edges. In a directed graph, a sink is a vertex that has every incident edge directed toward it, and a source is a vertex that has every incident edge directed outwards. In the alignment graph, $(0,0)$ is the unique source and (m,n) is the unique sink.

The alignment graph is very useful for understanding and developing alignment algorithms as we shall see in Chapter 3. One reason for this is that all possible alignments correspond one-to-one to the directed paths from the source $(0,0)$ to the sink (m,n). Hence, it gives a compact representation of all possible alignments of

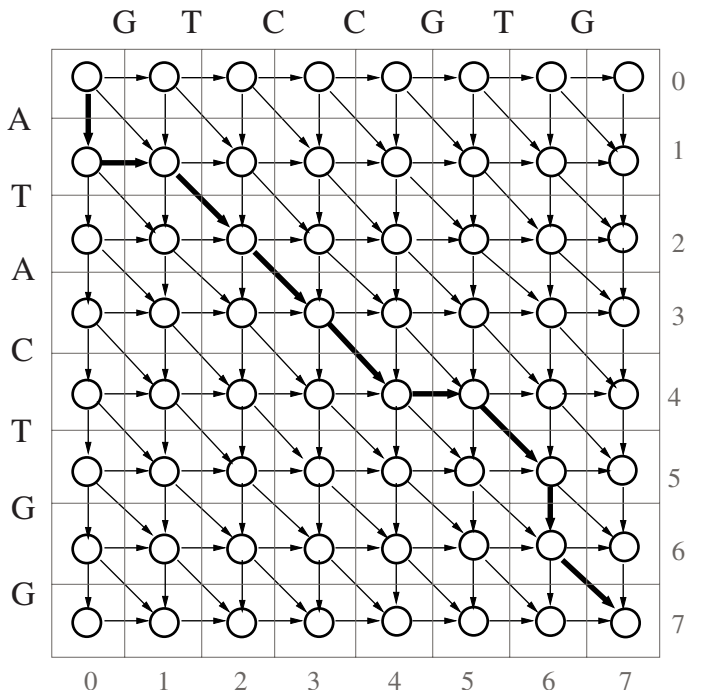

Fig. 1.2 Alignment graph for sequences ATACTGG and GTCCGTG.

the sequences. To see the correspondence, we consider the alignment graph given in Figure 1.2 in what follows.

Let $X = \texttt{ATACTGG}$ be the first sequence and $Y = \texttt{GTCCGTG}$ be the second sequence. The highlighted path in the alignment graph is

$$s \to (1,0) \to (1,1) \to (2,2) \to (3,3) \to (4,4) \to (4,5) \to (5,6) \to (6,6) \to (7,7),$$

where s is the source $(0,0)$. Let x_i (y_i) denote the ith letter of the first (second) sequence. Walking along the path, we write down a column $\binom{x_i}{y_j}$, $\binom{x_i}{-}$, and $\binom{-}{y_j}$ if we see a diagonal, vertical, and horizontal edge entering vertex (i,j), respectively. After reaching the sink $(7,7)$, we obtain the following nine-column alignment of X and Y:

$$\begin{array}{ll} X & \texttt{A - T A C - T G G} \\ Y & \texttt{- G T C C G T - G} \end{array}$$

where the ith column corresponds to the ith edge in the path.

On the other hand, given a k-column alignment of X and Y, we let u_i (and v_i) be the number of letters in the first i columns in the row of X (and Y) for $i = 0,1,\ldots,k$. Trivially, $u_0 = v_0 = 0$ and $u_k = m$ and $v_k = n$. For any i, we also have that

$$0 \leq u_{i+1} - u_i \leq 1$$

and

$$0 \leq v_{i+1} - v_i \leq 1.$$

This fact implies that there is an edge from (u_i, v_i) to (u_{i+1}, v_{i+1}). Thus,

$$(u_0, v_0) \rightarrow (u_1, v_1) \rightarrow \ldots \rightarrow (u_k, v_k)$$

is a path from the source to the sink. For example, from alignment

$$\begin{array}{ll} X & \texttt{A T A C T G - G} \\ Y & \texttt{G T C C - G T G,} \end{array}$$

we obtain the following path

$$(0,0) \rightarrow (1,1) \rightarrow (2,2) \rightarrow (3,3) \rightarrow (4,4) \rightarrow (5,4) \rightarrow (6,5) \rightarrow (6,6) \rightarrow (7,7).$$

In summary, an alignment of two sequences corresponds uniquely to a path from the source to the sink in the alignment graph of these two sequences. In this correspondence, match/mismatches, deletions, and insertions correspond to diagonal, vertical, and horizontal edges, respectively.

The aim of alignment is to find the best one of all the possible alignments of two sequences. Hence, a natural question to ask is: How many possible alignments are there for two sequences of length m and n? This is equivalent to asking how many different paths there are from the source node $(0,0)$ to the sink node (m,n) in an alignment graph. For a simple problem of aligning x_1x_2 and y_1, all 5 possible alignments are shown below:

$$\begin{array}{ccccc} x_1\ x_2 & x_1\ x_2 & \texttt{-}\ x_1\ x_2 & x_1\ x_2\ \texttt{-} & x_1\ \texttt{-}\ x_2 \\ y_1\ \texttt{-} & \texttt{-}\ y_1 & y_1\ \texttt{-}\ \texttt{-} & \texttt{-}\ \texttt{-}\ y_1 & \texttt{-}\ y_1\ \texttt{-} \end{array}$$

We denote the number of possible alignments by $a(m,n)$. In an alignment graph, the paths from $(0,0)$ to a vertex (i,j) correspond to possible alignments of two sequences of length i and j. Hence, $a(i,j)$ is equal to the number of paths from $(0,0)$ to the vertex (i,j). Because every path from the source $(0,0)$ to the sink (m,n) must go through exactly one of the vertices $(m-1,n)$, $(m,n-1)$, and $(n-1,m-1)$, we obtain

$$a(m,n) = a(m-1,n) + a(m,n-1) + a(n-1,m-1). \tag{1.1}$$

Noting that there is only one way to align the empty sequence to a non-empty sequence, we also have that

$$a(0,k) = a(k,0) = 1 \tag{1.2}$$

for any $k \geq 0$.

Table 1.1 The number of possible alignments of two sequences of length m and n for $n, m \leq 8$.

	0	1	2	3	4	5	6	7	8
0	1	1	1	1	1	1	1	1	1
1		3	5	7	9	11	13	15	17
2			13	25	41	61	85	113	145
3				63	129	231	377	575	833
4					321	681	1,289	2,241	3,649
5						1,683	3,653	7,183	13,073
6							8,989	19,825	40,081
7								48,639	108,545
8									265,729

Recurrence relation (1.1) and basis conditions (1.2) enable us to calculate $a(m,n)$ efficiently. Table 1.1 gives the values of $a(m,n)$ for any $m, n \leq 8$. As we see from the table, the number $a(m,n)$ grows quite fast with m and n. For $n = m = 8$, this number is already 265,729! As the matter of fact, we have that (see the gray box below)

$$a(m,n) = \sum_{k=\max\{n,m\}}^{m+n} \binom{k}{m} \binom{m}{n+m-k}. \tag{1.3}$$

In particular, when $m = n$, the sum of the last two terms of the right side of (1.3) is

$$\binom{2n-1}{n} \binom{n}{1} + \binom{2n}{n} = \frac{(n+2)(2n)!}{2(n!)^2}.$$

Applying Stirling's approximation

$$x! \approx \sqrt{2\pi} x^{x+1/2} e^{-x},$$

we obtain $a(100,100) > 2^{200} \approx 10^{60}$, an astronomically large number! Clearly, this shows that it is definitely not feasible to examine all possible alignments and that we need an efficient method for sequence alignment.

Proof of formula (1.3)

We first consider the arrangements of letters in the first row and then those in the second row of a k-column alignment. Assume there are m letters in the first row. There are $\binom{k}{m}$ ways to arrange the letters in the first row. Fix such an arbitrary arrangement. By the definition of alignment, there are $k-n$ spaces in the second row, and these spaces must be placed below the letters in the first row. Thus, there are $\binom{m}{k-n} = \binom{m}{m+n-k}$ arrangements for the second row. By the multiplication principle, there are

$$a_k(m,n) = \binom{k}{m}\binom{m}{m+n-k}$$

possible alignments of k columns.

Each alignment of the sequences has at least $\max\{m,n\}$ and at most $m+n$ columns. Summing $a_k(m,n)$ over all k from $\max\{m,n\}$ to $m+n$ yields formula (1.3).

1.3 Scoring Alignment

Our goal is to find, given two DNA or protein sequences, the best alignment of them. For this purpose, we need a rule to tell us the goodness of each possible alignment. The earliest similarity measure was based on percent identical residues, that is, simply to count matches in an alignment. In the old days, this simple rule worked because it was rare to see the low percent identity of two proteins with similar functions like homeodomain proteins. Nowadays, a more general rule has to be used to score an alignment.

First, we have a score $s(a,b)$ for matching a with b for any pair of letters a and b. Usually, $s(a,a) > 0$, and $s(a,b) < 0$ for $a \neq b$. Assume there are k letters $a_1, a_2, \ldots, a_k$. All these possible scores are specified by a matrix $(s(a_i,b_j))$, called a *scoring matrix*. For example, when DNA sequences are aligned, the following scoring matrix may be used:

A	2	-1	-1	-1
G	-1	2	-1	-1
C	-1	-1	2	-1
T	-1	-1	-1	2
	A	G	C	T

This scoring matrix indicates that all matches score 2 whereas mismatches are penalized by 1. Assume we align two sequences. If one sequence has letter a and another has b in a position, it is unknown whether a had been replaced by b or the other way around in evolutionary history. Thus, scoring matrices are usually symmetric like the one given above. In this book, we only write down the lower triangular part of a scoring matrix if it is symmetric.

Scoring matrices for DNA sequence alignment are usually simple. All different matches score the same, and so do all mismatches. For protein sequences, however, scoring matrices are quite complicated. Frequently used scoring matrices are developed using statistical analysis of real sequence data. They reflect the frequency of an amino acid replacing another in biologically related protein sequences. As a result, a scoring matrix for protein sequence alignment is usually called a *substitu-*

tion matrix. We will discuss in detail how substitution matrices are constructed and selected for sequence alignment in Chapter 8.

Second, we consider indels. In an alignment, indels and mismatches are introduced to bring up matches that appear later. Thus, indels are penalized like mismatches. The most straightforward method is to penalize each indel by some constant δ. However, two or more nucleotides are frequently inserted or deleted together as a result of biochemical processes such as replication slippage. Hence, penalizing a gap of length k by $-k\delta$ is too cruel. A *gap* in an alignment is defined as a sequence of spaces locating between two letters in one row. A popular gap penalty model, called *affine gap penalty*, scores a gap of length k as

$$-(o+k\times e),$$

where $o>0$ is considered as the penalty for opening a gap and $e>0$ is the penalty for extending a gap by one letter. The opening gap penalty o is usually big whereas the gap extension penalty e is small. Note that simply multiples of the number of indels is a special case of the affine gap penalty model in which $o=0$.

A scoring matrix and a gap penalty model form a *scoring scheme* or a *scoring system*. With a scoring scheme in hand, the score of an alignment is calculated as the sum of individual scores, one for each aligned pair of letters, and scores for gaps. Consider the comparison of two DNA sequences with the simple scoring matrix given above, which assigns 2 to each match and -1 to each mismatch. If we simply penalize each indel by -1.5, the score for the alignment on page 4 is

$$-1.5-1.5+2-1+2-1.5+2-1.5+2=3.$$

As we will see in Section 8.3, in any scoring matrix, the substitution score $s(a,b)$ is essentially a logarithm of the ratio of the probability that we expect to see a and b aligned in biologically related sequences to the probability that they are aligned in unrelated random sequences. Hence, being the sum of individual log-odds scores, the score of a ungapped alignment reflects the likelihood that this alignment was generated as a consequence of sequence evolution.

1.4 Computing Sequence Alignment

In this section, we briefly define the global and local alignment problems and then relate the alignment problem to some interesting algorithmic problems in computer science, mathematics, and information theory.

1.4.1 Global Alignment Problem

With a scoring system, we associate a score to each possible alignment. The *optimal alignments* are those with the maximum score. The *global alignment problem* (or simply alignment problem) is stated as

Global Alignment Problem

Input: Two sequences x and y and a scoring scheme.
Solution: An optimal alignment of x and y (as defined by the scoring scheme).

Because there is a huge number of possible alignments for two sequences, it is not feasible to find the optimal one by examining all alignments one by one. Fortunately, there is a very efficient algorithm for this problem. This algorithm is now called the Needleman-Wunsch algorithm. The so-called dynamic programming idea behind this algorithm is so simple that such an algorithm has been discovered and rediscovered in different form many times. The Needleman-Wunsch algorithm and its generalizations are extensively discussed in Chapter 3.

The sequence alignment problem seems quite simple. But, it is rather general as being closely related to several interesting problems in mathematics, computer science, and information theory. Here we just name two such examples. The *longest common subsequence problem* is, given two sequences, to find a longest sequence whose letters appear in each sequence in order, but not necessarily in consecutive positions. This problem had been interested in mathematics long before Needleman and Wunsch discovered their algorithm for aligning DNA sequences. Consider a special scoring system $\mathscr{S}$ that assigns 1 to each match, $-\infty$ to each mismatch, and 0 to each indel. It is easy to verify that the optimal alignment of two sequences found using $\mathscr{S}$ must not contain mismatches. As a result, all the matches in the alignment give a longest common subsequence. Hence, the longest common subsequence problem is identical to the sequence alignment problem under the particular scoring system $\mathscr{S}$.

There are two ways of measuring the similarity of two sequences: similarity scores and distance scores. In distance scores, the smaller the score, the more closely related are the two sequences. Hamming distance allows one only to compare sequences of the same length. In 1966, Levenshtein introduced *edit distance* for comparison of sequences of different lengths. It is defined as the minimum number of editing operations that are needed to transform one sequence into another, where the editing operations include insertion of a letter, deletion of a letter, and substitution of a letter for another. It is left to the reader to find out that calculating the edit distance between two strings is equivalent to the sequence alignment problem under a particular scoring system.

1.4.2 Local Alignment Problem

Proteins often have multiple functions. Two proteins that have a common function may be similar only in functional domain regions. For example, homeodomain proteins, which play important roles in developmental processes, are present in a variety of species. These proteins in different species are only similar in one domain of about 60 amino acids long, encoded by homeobox genes. Obviously, aligning the entire sequences will not be useful for identification of the similarity among homeodomain proteins. This raises the problem of finding, given two sequences, which respective segments have the best alignments. Such an alignment between some segments of each sequence is called *local alignment* of the given sequences. The problem of aligning locally sequences is formally stated as

Local Alignment Problem

Input: Two sequences $x = x_1x_2 \ldots x_m$ and $y = y_1y_2 \ldots y_n$ and a scoring scheme.
Solution: An alignment of fragments $x_ix_{i+1} \ldots x_j$ and $y_ky_{k+1} \ldots y_l$, that has the largest score among all alignments of all pairs of fragments of x and y.

A straightforward method for this problem is to find the optimal alignment for every pair of fragments of x and y using the Needleman-Wunsch algorithm. The sequence x has $\binom{m}{2}$ fragments and y has $\binom{n}{2}$ ones. Thus, this method is rather inefficient because its running time will increase by roughly m^2n^2 times. Instead, applying directly the dynamic programming idea leads to an algorithm that is as efficient as the Needleman-Wunsch algorithm although it is a bit more tricky this time. This dynamic programming algorithm, called *Smith-Waterman* algorithm, is covered in Section 3.4.

Homology search is one important application of the local alignment problem. In this case, we have a query sequence, say, a newly sequenced gene, and a database. We wish to search the entire database to find those sequences that match locally (to a significant degree) with the query. Because databases have easily millions of sequences, Smith-Waterman algorithm, having quadratic-time complexity, is too demanding in computational time for homology search. Accordingly, fast heuristic search tools have been developed in the past two decades. Chapter 4 will present several frequently used homology search tools.

Filtration is a useful idea for designing fast homology search programs. A filtration-based program first identifies short exact matches specified by a fixed pattern (called seed) of two sequences and then extends each match to both sides for local alignment. A clever technique in speeding up homology search process is to substitute optimized spaced seed for consecutive seed as exemplified in PatternHunter. Theoretic treatment of spaced seed technique is studied in Chapter 6.

1.5 Multiple Alignment

An alignment of two sequences is called an *pairwise* alignment. The above definition of pairwise alignment can be straightforwardly generalized to the case of multiple sequences. Formally, a *multiple alignment* of k sequences $X_1, X_2, \ldots, X_k$ over an alphabet Σ is specified by a $k \times n$ matrix M. Each entry of M is a letter of Σ or a space '-', and each row j contains the letters of sequence X_j in order, which may be interspersed with '-'s. We request that each column of the matrix contains at least one letter of Σ. Below is a multiple alignment of partial sequences of five globin proteins:

```
Hb_a    LSPADKTNVUAAWGKVGA----HAGEYGAE
Hb_b    LTPEEKSAVTALWGKV------NVDEVGGE
Mb_SW   LSEGEWQLVLHVWAKVEA----DVAGHGQD
LebHB   LTESQAALVKSSWEEFNA----NIPKHTHR
BacHB   QTINIIKATVPVLKEHG------V-TITTT
```

Multiple alignment is often used to assess sequence conservation of three or more closely related proteins. Biologically similar proteins may have very diverged sequences and hence may not exhibit a strong sequence similarity. Comparing many sequences at the same time often finds weak similarities that are invisible in pairwise sequence comparison.

Several issues arise in aligning multiple sequences. First, it is not obvious how to score a multiple alignment. Intuitively, high scores should correspond to highly conserved sequences. One popular scoring method is the *Sum-of-Pairs (SP) score*. Any alignment A of k sequences $x_1, x_2, \ldots, x_k$ gives a pairwise alignment $A(x_i, x_j)$ of x_i and x_j when restricted to these two sequences. We use $s(i, j)$ to denote the score of $A(x_i, x_j)$. The SP score of the alignment A is defined as

$$SP(A) = \sum_{1 \le i < j \le k} s(i, j).$$

Note that the SP score is identical to the score of a pairwise alignment when there are only two sequences. The details of the SP score and other scoring methods can be found in Section 5.2

Second, aligning multiple sequences is extremely time-consuming. The SP score of a multiple alignment is a generalization of a pairwise alignment score. Similarly, the dynamic programming algorithm can generalize to multiple sequences in a straightforward manner. However, such an algorithm will use roughly $(2m)^k$ arithmetic operations for aligning k sequences of length m. For small k, it works well. The running time is simply too much when k is large, say 30. Several heuristic approaches have been proposed for speeding up multiple alignment process (see Section 5.4).

1.6 What Alignments Are Meaningful?

Although homology and similarity are often interchanged in popular usage, they are completely different. Homology is qualitative, which means having a common ancestor. On the other hand, similarity refers to the degree of the match between two sequences. Similarity is an expected consequence of homology, but not a necessary one. It may occur due to chance or due to an evolutionary process whereby organisms independently evolve similar traits such as the wings of insect and bats.

Assume we find a good match for a newly sequenced gene through database search. Does this match reflect a homology? Nobody knows what really happened over evolutionary time. When we say that a sequence is homologous to another, we are stating what we believe. No matter how high is the alignment score, we can never be 100% sure. Hence, a central question in sequence comparison is how frequently an alignment score is expected to occur by chance. This question has been extensively investigated through the study of the alignments of random sequences. The Karlin-Altschul alignment statistics covered in Chapter 7 lay the foundation for answering this important question.

To approach theoretically the question, we need to model biological sequences. The simplest model for random sequences assumes that the letters in all positions are generated independently, with probability distribution

$$p_1, p_2, \ldots, p_r$$

for all letters $a_1, a_2, \ldots a_r$ in the alphabet, where r is 4 for DNA sequences and 20 for protein sequences. We call it the *Bernoulli sequence model*.

The theoretical studies covered in Chapters 6 and 7 are based on this simple model. However, most of the results generalize to the high-order Markov chain model in a straightforward manner. In the *kth-order Markov chain sequence model*, the probability that a letter is present at any position j depends on the letters in the preceding k sites: $i-k, i-k+1, \ldots, j+1$. The third-order Markov chain model is often used to model gene coding sequences.

1.7 Overview of the Book

This book is structured into two parts. The first part examines alignment algorithms and techniques and is composed of four chapters, and the second part focuses on the theoretical issues of sequence comparison and has three chapters. The individual chapters cover topics as follows.

2 Basic algorithmic techniques. Starting with basic definitions and notions, we introduce the greedy, divide-and-conquer, and dynamic programming approaches that are frequently used in designing algorithms in bioinformatics.

3 Pairwise sequence alignment. We start with the dot matrix representation of pairwise alignment. We introduce the Needleman-Wunsch and Smith-Waterman algorithms. We further describe several variants of these two classic algorithms for coping with special cases of scoring schemes, as well as space-saving strategies for aligning long sequences. We also cover constrained sequence alignment and suboptimal alignment.

4 Homology search tools. After showing how filtration technique speeds up homology search process, we describe in detail four frequently used homology search tools: FASTA, BLAST, BLAT, and PatternHunter.

5 Multiple sequence alignment. Multiple sequence alignment finds applications in prediction of protein functions and phylogenetic studies. After introducing the sum-of-pairs score, we generalize the dynamic programming idea to aligning multiple sequences and describe how progressive approach speeds up the multiple alignment process.

6 Anatomy of spaced seeds. We focus on the theoretic analysis of spaced seed technique. Starting with a brief introduction to the spaced seed technique, we first discuss the trade-off between sensitivity and specificity of seeding-based methods for homology search. We then present a framework for the analysis of the hit probability of spaced seeds and address seed selection issues.

7 Local alignment statistics. We focus on the Karlin-Altschul statistics of local alignment scores. We show that optimal segment scores are accurately described by an extreme value distribution in asymptotic limit, and introduce the Karlin-Altschul sum statistic. In the case of gapped local alignment, we describe how the statistical parameters for the score distribution are estimated through empirical approach, and discuss the edge-effect and multiple testing issues. Finally, we illustrate how the Expect value and P-value are calculated in BLAST using two BLAST printouts.

8 Scoring matrices. We start with the frequently used PAM and BLOSUM matrices. We show that scoring matrices for aligning protein sequences take essentially a log-odds form and there is one-to-one correspondence between so-called valid scoring matrices and the sets of target and background frequencies. We also discuss how scoring matrices are selected and adjusted for comparing sequences of biased letter composition. Finally, we discuss gap score schemes.

1.8 Bibliographic Notes and Further Reading

After nearly 50 years of research, there are hundreds of available tools and thousands of research papers in sequence alignment. We will not attempt to cover all (or even a large portion) of this research in this text. Rather, we will be content to provide

pointers to some of the most relevant and useful references on the topics not covered in this text.

For the earlier phase of sequence comparison, we refer the reader to the paper of Sankoff [174]. For information on probabilistic and statistical approach to sequence alignment, we refer the reader to the books by Durbin et al. [61], Ewens and Grant [64], and Waterman [197]. For information on sequence comparisons in DNA sequence assembly, we refer the reader to the survey paper of Myers [149] and the books of Gusfield [85] and Deonier, Tavaré and Waterman [58]. For information on future directions and challenging problems in comparison of genomic DNA sequences, we refer the reader to the review papers by Batzoglou [23] and Miller [138].