The Performance of Sequence Alignment Algorithms

Leila Alimehr
Abstract

The Performance of Sequence Alignment Algorithms

Leila Alimehr

This thesis deals with sequence alignment algorithms. The sequence alignment is a mutual arrange of two or more sequences in order to study their similarity and dissimilarity. Four decades after the seminal work by Needleman and Wunsch in 1970, these methods still need more explorations.

We start out with a review of a sequence alignment, and its generalization to multiple alignments, although the focus of this thesis is on the evaluation of the new alignment algorithms. The research presented herein has stepped into the different algorithms that are in terms of the dynamic programming. In the study of sequence alignment algorithms, two powerful techniques have been invented. According to the simulations, the new algorithms are shown to be extremely efficient for the comparing DNA sequences. All the sequence alignment algorithms are compared in terms of the distance. We use the programming language R for the implementation and simulation of the algorithms discussed in this thesis.

Key words: Algorithms; Dynamic programming; Genetics; Sequence alignment.
Acknowledgements

First I would like to thank, Dr. Saeid Amiri, for the patient guidance, encouragement and advice he has provided throughout my time as his student. I should also thank Prof. Dietrich von Rosen, for supporting me during my master thesis in biometry section in SLU.

Next, I should thank Dr. Mohamed Faouzi Atig, who agreed to review my master thesis. I would like to thank my beautiful parents, Golafrooz and Houshang, for providing me with the support needed in order to continually push myself to succeed. Without their love and support, I would not be here today. Thank you mother for always pushing me.

I would like to thank my sister Neda, for being my friend and I will love you forever.

I like to thank Mr Nikooei, for your live that you show me regularly. You are generous with your time, energy and forgiveness.

I specially wish to express my love for Yalda, my darling sister, who helps me during my education and guide and support me throughout the years. I am so grateful for everything that she has given me and I could not have done it without her. I thank her from the bottom of my heart for her patience, understanding, guidance and endless love that she always has given to me.
Dedicated to all who have been making my life pleasant

My parents, Yalda and Mr Nikooei
# Contents

## Acknowledgements

<table>
<thead>
<tr>
<th>1 Introduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Sequence Alignment</td>
</tr>
<tr>
<td>3 Pairwise Sequence Alignments</td>
</tr>
<tr>
<td>4 New Sequence Alignments</td>
</tr>
</tbody>
</table>

### 2 Sequence Alignment

- 0.1 **Sequence Alignment**
- 0.2 **Scoring Alignment**
- 0.3 **Distance**
- 0.4 **Dynamic Programming**

### 3 Pairwise Sequence Alignments

- 0.1 **Global Alignment: Needleman-Wunsch Algorithm**
  - 0.1.1 **Initialization Step**
  - 0.1.2 **Matrix Filling Step**
  - 0.1.3 **Traceback Step**
  - 0.1.4 **Global Alignment Algorithms**
- 0.2 **Local Alignment: Smith-Waterman Algorithm**
  - 0.2.1 **Initialization Step**
  - 0.2.2 **Matrix Filling Step**
  - 0.2.3 **Traceback Step**
  - 0.2.4 **Local Alignment Algorithms**
- 0.3 **Semi-Global Alignment**
  - 0.3.1 **Initialization Step**
  - 0.3.2 **Matrix Filling Step**
  - 0.3.3 **Traceback Step**
  - 0.3.4 **Semi-Global Algorithms**

### 4 New Sequence Alignments

- 0.1 **Weighted Alignment**
  - 0.1.1 **Initialization Step**
  - 0.1.2 **Matrix Filling Step**
  - 0.1.3 **Traceback Step**
  - 0.1.4 **Weighted Alignment Algorithms**
- 0.2 **Diagonal Alignment**
  - 0.2.1 **Initialization Step**
4.2.2 Matrix Filling Step ................................................. 37
4.2.3 Traceback Step ..................................................... 37
4.2.4 Diagonal Alignment Algorithms ......................... 39

5 Simulation

5.1 Homogeneous ....................................................... 43
5.2 Non-homogeneous .................................................. 44

6 Conclusion and Future Work ............................. 50

Bibliography ............................................................ 51
Chapter 1

Introduction

Due to the importance of the DNA’s linear structure and protein, the sequence comparison has become one of the major aspects in molecular biology since the first protein sequence was read in the 1950s. The use of sequence comparison is motivated by different reasons. Among these reasons are: the sequence comparison enables identification of genes and preserved sequence patterns; they can be used to form structural, functional and evolutionary relationships between the DNA and proteins; between different species they give a reliable method for inducing the biological functions for new formed sequenced gene.

The study of biological sequence data can reveal that genes in an organism have significant similarity with those in other organisms that diverged from many years ago. It should be noted that the organisms are different in some aspects due of the evolution of the specie genomes. Therefore the comparison of related genomes helps to understand the DNA language and draws inference about the evolutionary relationship between the different species.

Two sequences from different genomes are homologous if they evolved from a common ancestor, hence the exploration of them is important because they usually have preserved functions and structures. Since species diverged at different periods of times, the homologous DNA or proteins are expected to be more similar in closely related organisms than in remotely related ones. Usually scientists have little idea about the function of new formed proteins and by study in database can find similarity in their homologies. We consider the algorithms for sequence comparisons between the DNA molecules in this work. For more information about genetics and related subjects see, for example
The source data of sequence is widely available, see for example [1], [2] and [9].

The sequence alignment is used to determine the similarity between two sequences over some alphabet. In this thesis, we use a finite set of alphabet where they are the chemical components of DNA, Adenine (A), Cytosine (C), Guanine (G), and Thymine (T). Our aim is to find the best alignment between two DNA sequences. For this aim, we use a scoring alignment. Consider the sequences $GCAGG$ and $AAAGTGG$ with different lengths. After aligning these sequences, we obtain two sequences with the same length. So we have:

$$
\begin{array}{cccccccc}
G & C & A & \phi & G & \phi & G & \phi \\
\phi & A & A & A & G & T & G & G \\
\end{array}
$$

The above alignment includes two rows and eight columns. If the alphabets in the column are the same, it is named matches. If the alphabets in the column are different, it is named mismatches, and the column includes $\phi$, named indels. The above alignment includes three matches (3th, 5th and 7th columns), one mismatch (2th column) and four indels (1th, 4th, 6th and 8th columns). The scoring is simple. All the different matches and mismatches score the same and all the indels are penalized by constant. The score of an alignment is the sum of the scores of all columns with no $\phi$ minus the gap penalties.

Very short sequences might be aligned by hand. However, interesting problem is the alignment of extremely numerous sequences that cannot be aligned by human efforts. Instead, human knowledge is used in constructing algorithms to produce high-quality sequence alignment. Computational approaches to sequence alignment generally fall into two categories: global alignment and local alignment. The global alignment by definition is to align sequences from end to end. Local alignment are intended for comparing sequences containing locally similar region and semi-global alignment is similar to the global alignment, the differences between them are the way for scoring alignments and choosing the maximum in the last row or column to traceback.

Despite the existence of a few different algorithms for the global sequence alignment, there is still a need for a better method. This thesis explores new approaches to carry out the sequence alignment. The global method considers the equal cost to the penalty of indel. The main goal of this thesis is finding alignments according to the global alignment that adopt appropriate cost according to the movement or position. For this purpose, we define two new alignments; weighted and diagonal, first one gives weights
according to the position and direction and the second one gives different costs accord-
ing to the position in the matrix alignment. We compare the performance of existing
alignment with new alignments under the same ancestor and different ancestor. Under
the same ancestor, the new alignment algorithms are working closely to the global align-
ment, but under the different ancestor, the two new algorithms are working much better
than global alignment.

The presentation of this thesis is as follows. Chapter 2 explains the sequence alignment
concepts and techniques. Chapter 3 presents the existing algorithms to carry out the
sequence alignment. Two new algorithms are outlined in Chapter 4. Chapter 5 includes
the simulations to compare the performance of the proposed sequences under same
ancestor and different ancestor. Chapter 6 presents the conclusion and gives the direction
of future work.
Chapter 2

Sequence Alignment

In this chapter, we recall some basic concepts and definitions such as: sequence alignment, distance and dynamic programming. For more informations the reader is referred to [4], [5], [8] and [13].

2.1 Sequence Alignment

The sequence alignment is used to determine the similarity between two sequences over some alphabet. An alphabet $\Sigma$ is a finite set of symbols. In this thesis we will use $\Sigma = \{A, T, C, G\}$. A sequence over an alphabet is a finite ordered set of symbols from the alphabet. We will use the following notation to represent the sequences $X = x_1, \ldots, x_m$ and $Y = y_1, \ldots, y_n$ over $\Sigma$. For simplicity, let us assume without loss of generality that $m \leq n$ in the rest of this thesis. We extend $\Sigma$ by adding a special symbol $\phi$, which denotes null characters or nulls for short, some authors use $\lambda$ or "-" or blank instead $\phi$. An alignment (as defined in [4]) of two sequences $X$ and $Y$ is a two-row matrix with entries from $X$ and $Y$, and possibly interspersed with nulls such that:

1. The first (second) row contains the letters of $X$ ($Y$) in order;
2. One or more nulls $\phi$ may appear between two consecutive letters of $\Sigma$ in each row;
3. Each column contains at least one letter of $\Sigma$.

An alignment of two sequences poses an hypothesis concerning the evolution of the two sequences from the closest common ancestor. In an alignment we have three elements; matches, mismatches and indels. Columns are called matches if the letters are equal and mismatches otherwise. Mismatches show a substitution event when a nucleotide is replaced by another in the sequence. Indels are columns with null. Columns with null in the first row are called insertions that represents that one or several nucleotides are inserted in the sequence, we denote it with $(\phi y_j)$. Columns with null in the second
row are called deletions, that represents one or several nucleotides are deleted in the sequence, it is denoted by \( (x^i_\phi) \). Consider two sequences GCAGT and AATCGAA, a possible alignment of them can be:

\[
\begin{align*}
G & \quad C & \quad A & \quad \phi & \quad \phi & \quad G & \quad \phi & \quad T \\
\phi & \quad A & \quad A & \quad T & \quad C & \quad G & \quad A & \quad A
\end{align*}
\]

There are one deletion (1st column), three insertions (4th, 5th and 7th columns), two mismatches (2nd and 8th columns) and two matches (3rd and 6th columns). The obtained sequences after alignment are:

\[
X^* = \{x^*_1, \ldots, x^*_L\},
\]
\[
Y^* = \{y^*_1, \ldots, y^*_L\},
\]

where

\[
\max\{n, m\} \leq L \leq n + m.
\]

The maximum length of alignment can occur in the following situation:

\[
\begin{align*}
x_1 & \ldots \quad x_m & \quad \phi & \quad \ldots & \quad \phi \\
\phi & \quad \ldots & \quad \phi & \quad y_1 & \ldots & \quad y_n
\end{align*}
\]

In order to see the number of alignments, let us define \( f(m, n) \) as the number of alignments of two sequences with \( n \) and \( m \) letters. The proof is trivial: if \( x_m \) is deleted then there are \( f(m - 1, n) \) alignments, if \( x_m \) and \( y_n \) are included then there are \( f(m - 1, n - 1) \) alignments, if \( y_n \) deleted then there are \( f(m, n - 1) \) alignments, hence:

\[
f(m, n) = f(m - 1, n) + f(m - 1, n - 1) + f(m, n - 1).
\]

Obviously there is one possibility to align the empty sequence to non-empty sequence, hence:

\[
f(m, \phi) = f(\phi, n) = 1.
\]

For \( n \geq 1 \), \( f(m, n) \) can be obtained using a simple algorithm, although its closed form exists,

\[
f(m, n) = \sum_{k=\max\{n, m\}}^{n+m} C(k, m)C(m, n + m - k),
\]

where \( C(k, m) \) is the binomial coefficients

\[
C(k, m) = \frac{k!}{m!(k-m)!}.
\]
In order to prove, let us consider the arrangement of \( m \) letters in the first row and \( n \) letters in the second row, that should fulfill \( k \) columns. The number of arrangements of \( m \) letters in \( k \) columns of the first row is \( \binom{k}{m} \), let us fix an arbitrary arrangement:

\[
y_1 \ldots y_m \phi_1 \ldots \phi_{k-m}
\]

In order to have an alignment with length \( k \), \( k - m \) elements of \( n \) letters should be placed in the second row exactly under the \( \phi_1 \ldots \phi_{k-m} \) so we have:

\[
y_1 \ldots y_m \phi_1 \ldots \phi_{k-m} \quad x_{n-(k-m)} \ldots x_n
\]

Hence the rest of elements, \( n - (k - m) \) of \( n \) elements, must be placed below the letters of the first row. Therefore the number of arrangements of such situation is \( \binom{m}{k} \binom{n}{m-(k-m)} \) arrangements, each alignment has at least \( \max\{n,m\} \) and at most \( n + m \) elements, hence easily can establish it. Under \( n = m \),

\[
f(n, n) = \sum_{k=n}^{2n} C(k, n) C(n, 2n - k) > C(2n - 1, n) C(n, 1) + C(2n, n) = \frac{(n + 2)(2n)!}{2(n!)^2},
\]

easily can show \( f(10, 10) > 10^{60} \), that is an astronomical number and such a number of possible alignments is not trivial to handle.

### 2.2 Scoring Alignment

In the alignment of two DNA sequences, the purpose is to find the best alignment. For this goal, we need a way to tell us about the goodness of each possible alignment. The earliest similarity was based on the number of matches in an alignment. Nowadays, a more general rule has been used for the scoring alignment.

First, we have a score \( \delta(x_i, y_j) \) for matching \( x_i \in \Sigma \) and \( y_j \in \Sigma \) for \( i \in \{1, \ldots, m\} \) and \( j \in \{1, \ldots, n\} \). Often \( \delta(x_i, x_i) > 0 \) and \( \delta(x_i, y_j) < 0 \) for \( x_i \neq y_j \). All the possible scores are specified with a matrix \( (\delta(x_i, y_j)) \), called a scoring matrix. The elements of the scoring matrix are as below:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>+\alpha</td>
<td>-\gamma</td>
<td>-\gamma</td>
<td>-\gamma</td>
</tr>
<tr>
<td>C</td>
<td>-\gamma</td>
<td>+\alpha</td>
<td>-\gamma</td>
<td>-\gamma</td>
</tr>
<tr>
<td>G</td>
<td>-\gamma</td>
<td>-\gamma</td>
<td>+\alpha</td>
<td>-\gamma</td>
</tr>
<tr>
<td>T</td>
<td>-\gamma</td>
<td>-\gamma</td>
<td>-\gamma</td>
<td>+\alpha</td>
</tr>
</tbody>
</table>
The scoring matrix shows that all matches score $\alpha$ and all mismatches are penalized by $\gamma$. Scoring matrices for DNA sequence alignments are usually simple. All different matches and mismatches score the same. Let consider that the matches score 4 whereas the mismatches are penalized by 2. Therefore the scoring matrix can be in the following form:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>+4</td>
<td>-2</td>
<td>-2</td>
<td>-2</td>
</tr>
<tr>
<td>C</td>
<td>-2</td>
<td>+4</td>
<td>-2</td>
<td>-2</td>
</tr>
<tr>
<td>G</td>
<td>-2</td>
<td>-2</td>
<td>+4</td>
<td>-2</td>
</tr>
<tr>
<td>T</td>
<td>-2</td>
<td>-2</td>
<td>-2</td>
<td>+4</td>
</tr>
</tbody>
</table>

Against the discussed approach, that gives the equal mismatch penalty, the scoring matrix with unequal penalties can be adopted. For instance, in the study of mutation in homologous genes, the transition mutation ($A \rightarrow G$, $G \rightarrow A$, $C \rightarrow T$, $T \rightarrow C$) occurs approximately twice as frequently as do transversion ($A \rightarrow T$, $T \rightarrow A$, $A \rightarrow C$, $C \rightarrow T$, etc). Hence it makes sense to assign a lower penalty for transitions than transversions, [5]. The scoring matrix can be of the following form:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>$\alpha$</td>
<td>$-\gamma$</td>
<td>$-\gamma$</td>
<td>$-\gamma$</td>
</tr>
<tr>
<td>C</td>
<td>$-\gamma$</td>
<td>$\alpha$</td>
<td>$-\gamma$</td>
<td>$-\gamma$</td>
</tr>
<tr>
<td>G</td>
<td>$-\gamma$</td>
<td>$-\gamma$</td>
<td>$\alpha$</td>
<td>$-\gamma$</td>
</tr>
<tr>
<td>T</td>
<td>$-\gamma$</td>
<td>$-\gamma$</td>
<td>$-\gamma$</td>
<td>$\alpha$</td>
</tr>
</tbody>
</table>

The other approach is to use protein scoring for the DNA, [4].

### 2.3 Distance

In order to have a metric measure, the following properties are needed:

1. $d(s, t) \geq 0$.
2. $d(s, t) = 0$ iff $s \neq t$.
3. $d(s, t) = d(t, s)$.
4. $d(s, t) \leq d(s, u) + d(u, t)$.

Finding such a metric measure is not trivial. Define $d(x, y)$ as a distance on $(x, y)$ and $\beta$ be the positive cost of insertion or deletion. The distance between $X$ and $Y$ is defined as:

$$D(X, Y) = \min \sum_{i=1}^{\ell} d(x_i, y_i^*)$$
where the minimum is over all alignments of X and Y. This value can be found using a recursive process. Let:

$$D_{i,j} = D(x, \ldots, x_m, y_1, \ldots, y_n),$$

$$D_{0,0} = 0, \ D_{0,j} = \sum_{k=1}^{j} d(\phi, y_k), \ D_{i0} = \sum_{k=1}^{i} d(x_k, \phi),$$

then

$$D_{ij} = \min\{D_{i-1,j} + d(x_i, \phi), D_{i-1,j-1} + d(x_i, y_j), D_{i,j-1} + d(\phi, y_j)\}.$$  

If \(d(\cdot, \cdot)\) is a metric then \(D(\cdot, \cdot)\) is a metric on the set of finite sequences. In order to show the statement, clearly the sequence \(x_1, \ldots, x_i\) and \(y_1, \ldots, y_j\) can be in the following forms:

$$\ldots x_i \ldots x_i \ldots \phi \ldots \phi \ldots y_j \ldots y_j$$  

(2.2)

If the optimal alignment leads to \( (x_i^*) \), the cost is \( D_{i-1,j} + d(x_i, \phi) \). If the alignment ends to \( (y_j^*) \) the cost is \( D_{i-1,j-1} + d(x_i, y_j) \). The same discussion is held for \( (\phi^*) \).

Since the sequences have different lengths, one needs to use an appropriate measure to quantify the similarity of the two sequences. A similarity measure is a function that associates a number value to a pair of sequences such that a higher value indicates greater similarity and vice versa.

Similarity is used more common than distance because the problems that can be solved using the distance method, also can be solved with a similarity based method, [12]. In order to find a similarity, let us define \( S(X,Y) \) to be a similarity measure such that \( \delta(x_i, x_i) > 0 \) for \( x_i \in \Sigma \) and \( \delta(x_i, y_j) < 0 \) for \( x_i \neq y_j \ (i \in \{1, \ldots, m\} \text{ and } j \in \{1, \ldots, n\}) \).

The similarity of X and Y may be defined by:

$$S(X,Y) = \max\sum_{i=1}^{L} \delta(x_i^*, y_i^*),$$

where the maximum is over all alignments.

Using the same discussion given for the distance, the similarity leads to:

$$S(X,Y) = \max\{S_{i-1,j} - \delta(x_i, \phi), S_{i-1,j-1} + \delta(x_i, y_j), S_{i,j-1} - \delta(\phi, y_j)\}.$$  

see Theorem 9.8 in [12].

After the alignment, the similarity can be calculated by:
\[ S = \alpha N_1 - \gamma N_2 - \beta N_3, \]  
\hspace{1cm} (2.3)

where \( N_1, N_2 \) and \( N_3 \) are the number of matches, mismatches and indels, respectively.

## 2.4 Dynamic Programming

In bioinformatics, we use a sequence alignment for arranging the first sequence of DNA to recognize the similar regions that may cause functional, structural and evolutionary relationships among the sequences. The aligned sequences of nucleotide or amino acid residues are showed as rows in a matrix. Gaps are inserted among the residues with identical characters aligned in successive columns. Sequence alignment can be used for the non-biological sequences, for example distinguishing similarities in a series of letters and words used in a human language.

The short sequences can be aligned by hand, but we are interested in problems which need the alignment of lengthy, highly variable or extremely numerous sequences. Totally we have two categories for sequence alignments: the global alignment and local alignment.

Global alignment is a method of global optimization that forces the alignment to traverse the total length of query sequences. Local alignment recognize similar regions with long sequences which are usually distinct overall. We use a variety of computational algorithms for sequence alignment problems, including slow but formally optimal methods such as dynamic programming method.

Dynamic programming is a technique that divides the problem into a number of subproblems and solves every subproblem just once then saves its answer in a table, and therefore avoiding to recompute the answer for each subproblem, every time when it is encountered.

Dynamic programming is used usually for optimization problems. In the optimization problems, there exist many possible solutions, each solution has a value but we want to find a solution with an optimal value (minimum or maximum). We name such a solution as an optimal solution to the problem.

Dynamic programming can be used for global alignment via the Needleman-Wunsch algorithm ([10]) and local alignment via the Smith-Waterman algorithm ([11]). For DNA sequence alignments, we use a scoring matrix, and simply assign a positive match score, a negative mismatch score and a negative gap penalty. Dynamic programming includes three steps for DNA sequence alignments which are:

1) Initialization.
2) Matrix Filling (Scoring).

3) Traceback (Alignment).

In initialization step, we align two sequences across the columns and rows. An alignment can also start with a gap in one of the sequences. For example we are interested to align the two sequences GCAGG and AAAGTGG. The length of these sequences are 5 and 7, respectively. It is possible to start an alignment with a gap so the size of the matrix is $6 \times 8$. Initialization step is shown in Figure 2.1. We will use this example in the rest of this thesis.

In the matrix filling step, according to the alignment (global or local alignment), we fill the matrix. In the traceback step, we consider the path and move along it according to the dynamic programming. Chapter four explains these steps in details.

The discussion of alignment and Distance/Similarity using the statistical approach can be found in [12].

![Figure 2.1: Initialization Step in Dynamic Programming.](image)
Chapter 3

Pairwise Sequence Alignments

This chapter introduces the existing methods to carry out the sequence alignment.

3.1 Global Alignment: Needleman-Wunsch Algorithm

Needleman and Wunsch ([10]) described a dynamic programming algorithm for the global sequence alignment. By global sequence alignment, we mean that align both sequences globally and consider the whole of sequences. Consider two sequences $X = x_1, x_2, ..., x_m$ and $Y = y_1, y_2, ..., y_n$, an alignment between $X$ and $Y$ is gained by introducing $\phi$ into the two sequences such that the length of two resulting sequences are equal and no column contains two $\phi$. Let $\sum$ be the alphabet of the two sequences $X$ and $Y$. We use a very simple scoring schema as follows:

A score $\delta(x,y)$ is defined for each $(x_i, y_j) \in \sum \times \sum$, each indel (a column with null) is penalized by a constant $\beta$.

The score of an alignment is the sum of $\delta$ scores of all columns with no $\phi$ minus the gaps penalties. An optimal global alignment is an alignment that maximizes the score and considers a subset of the sequences. An amount of match, mismatch and gap are arbitrary. We consider the following scores, a match is given a bonus score $+4$, a mismatch is penalized by score $-2$ and the gap penalty for each indel is $1$. So we have:

$$\delta(x_i, y_j) = \begin{cases} +4 & \text{if } x_i = y_j, \\ -2 & \text{if } x_i \neq y_j. \end{cases}$$

For the gap penalty we have $\beta = 1$.

The alignment graph of $X$ and $Y$ is directed acyclic graph which vertices are the pairs $(i, j)$ where $i \in \{1, ..., m\}$ and $j \in \{1, ..., n\}$. These vertices are arrayed in $m + 1$ rows.
and \( n + 1 \) columns. The edge set consists of three types of edges. The substitution aligned pairs correspond to diagonal edge from \((i - 1, j - 1)\) to \((i, j)\), the deletion aligned pairs correspond to vertical edge from \((i - 1, j)\) to \((i, j)\) and the insertion aligned pairs correspond to horizontal edge from \((i, j - 1)\) to \((i, j)\). The procedure of the pairwise alignment can be done using the graph or the matrix, since the matrix schema is simpler to explain, matrix schema is considered here. We denote this method as GA in the rest of this thesis. Let us again consider our example in the part of dynamic programming and use the three steps to carry out the alignment.

3.1.1 Initialization Step

In this step each row \( S[i, 0] \) is set to \(-\beta \times i\), each column \( S[0, j] \) is set to \(-\beta \times j\). Recall that \( \beta \) is a gap penalty. So we have:

\[
S[0, 0] = 0, \quad S[i, 0] = -\beta \times i, \quad S[0, j] = -\beta \times j.
\]

Using this scoring schema, the initialization step is presented in Figure 3.1.

![Figure 3.1: Initialization Step in GA.](image)

3.1.2 Matrix Filling Step

To find the maximum global alignment score, start from the upper left hand corner of the matrix and finding the maximal score \( S[i, j] \) for each position.

Figure 3.2 presents three possible ways to enter to the grid point \((i, j)\), and one should choose the maximum of their path weights. The weight of maximum-scoring path entering \((i, j)\) from \((i - 1, j)\) vertically is the weight of maximum-scoring path entering
Figure 3.2: The ways entering to the point \((i,j)\) in GA.

\((i-1,j)\) plus the weight of edge \((i-1,j) \rightarrow (i,j)\). Therefore the weight of maximum-scoring path entering \((i,j)\) with a deletion gap symbol at the end is \(S[i-1,j] - \beta\).

Similarly weight of maximum-scoring path entering \((i,j)\) from \((i,j-1)\) horizontally is the weight of maximum-scoring path entering \((i,j-1)\) plus the weight of \((i,j-1) \rightarrow (i,j)\). Therefore the weight of maximum-scoring path entering \((i,j)\) with an insertion gap symbol at the end is \(S[i,j-1] - \beta\).

Finally the weight of maximum-scoring path entering \((i,j)\) from \((i-1,j-1)\) diagonally is \(S[i-1,j-1] + \delta(x_i,y_j)\).

We simply take the maximum value of these three choices to compute \(S[i,j]\).

\[
S[i,j] = \max\{S[i-1,j] - \beta, S[i,j-1] - \beta, S[i-1,j-1] + \delta(x_i,y_j)\}.
\]

The value \(S[m,n]\) is the score of an optimal global alignment between \(X\) and \(Y\). Using the above information, the score at position \(S[1,1]\) in the matrix can be calculated as:
\[
S[1,1] = \max\{S[0,0] - 2, S[1,0] - 1, S[0,1] - 1\} = \max\{-2, -2, -2\} = -2.
\]

The value of -2 is then placed in position \(S[1,1]\). Now we proceed to \(S[1,2] = \max\{S[0,1] - 2, S[1,1] - 2, S[0,2] - 1\} = \max\{-3, -3, -3\} = -3\). The procedure should be continued until filling the rest of the matrix. The resulting matrix is given in Figure 3.3.

### 3.1.3 Traceback Step

After filling the matrix, the maximum global alignment for the sequences is 6. The traceback step will give the actual alignments that result in the maximum score. The traceback begins in position \(S[m,n]\) and it will end in the position \(S[0,0]\). At each cell,
we look to see where should move next, according to the pointers. Figure 3.4 shows the two existing paths.

Initially, the possible predecessors are diagonal or horizontal. We start from the last cell $S[5, 7]$ until we reach the first cell $S[0, 0]$. In the upper path, $S[5, 7] = S[4, 6] + 4 = 2 + 4 = 6$, we make a diagonal move back to the entry $(4, 6)$, it corresponds to the substitution $(G\ G)$ in the alignment. Since $S[4, 6] = S[3, 5] + 4 = -2 + 4 = 2$, we make a diagonal move back to the entry $(3, 5)$, it corresponds to the substitution $(G\ G)$ in the alignment. Since $S[3, 5] = S[2, 4] - 2 = 0 - 2 = -2$, we make a diagonal move back to the entry $(2, 4)$, it corresponds to the insertion $(\phi\ G)$ in the alignment. Since $S[2, 4] = S[1, 4] - 1 = 1 - 1 = 0$, we make a vertical move back to the entry $(1, 4)$, it corresponds to the deletion $(C)$ in the alignment. We continue the path until reaching $S[0, 0]$.

In the lower path, $S[5, 7] = S[5, 6] - 1 = 7 - 1 = 6$, we make an horizontal move back to the entry $(5, 6)$, it corresponds to the insertion $(\phi\ G)$ in the alignment. Since $S[5, 6] = S[4, 5] + 4 = 3 + 4 = 7$, should make a diagonal move back to the entry $(4, 5)$, it corresponds to the substitution $(G\ G)$ in the alignment. Since $S[4, 5] = S[4, 4] - 1 = 4 - 1 = 3$, we make an horizontal move back to the entry $(4, 4)$, it corresponds to the insertion $(T)$ in the alignment. Since $S[4, 4] = S[3, 3] + 4 = 0 + 4 = 4$, we make a diagonal move back to the entry $(3, 3)$, it corresponds to the substitution $(G\ G)$ in the alignment. We continue the path until reaching $S[0, 0]$.

The resulting global alignments can be one of the following paths:

1) Upper path:

$$
\phi \ \phi \ \phi \ G \ C \ A \ G \ G \\
A \ A \ A \ G \ \phi \ T \ G \ G \\
- \ - \ - \ + \ - \ - \ + \ + \\
1 \ 1 \ 1 \ 4 \ 1 \ 2 \ 4 \ 4 
$$
The score of an alignment is the sum of $\delta$ scores of all columns with no $\phi$ minus the gaps penalties. Above alignment includes three matches, one mismatch and four indels. The score of match is $+4$, the score of mismatch is $-2$ and the gap penalty is $+1$, therefore the score of the alignment is:

$$3 \times (+4) + 1 \times (-2) - 4 \times (+1) = 12 - 2 - 4 = 6$$

2) Lower Path:

$$\begin{array}{cccccccc}
G & C & A & G & \phi & G & \phi \\
A & A & A & G & T & G & G \\
- & - & + & + & - & + & - \\
2 & 2 & 4 & 4 & 1 & 4 & 1
\end{array}$$

Above alignment includes three matches, two mismatches and two indels. Hence, the score of alignment is:

$$3 \times (+4) + 2 \times (-2) - 2 \times (+1) = 12 - 4 - 2 = 6$$

It should be mentioned that the different alignments might lead the same score. Thus the alignment is not unique.

Filling the table of sequences with $m$ and $n$ letters, one needs to calculate the elements of the first row and the first column that need $n + m$ computations. The rest cells, $m \times n$, need to compute three scores and take their maximum. Therefore for each cell, we have $4nm + n + m$ iterations. Hence, the complexity of the algorithm is $O(mn)$. Then constructing a single alignment can be done in $O(n + m)$ times, therefore the total complexity is $O(mn)$.

### 3.1.4 Global Alignment Algorithms

Figure 3.5 gives the pseudo-code for computing the score of an optimal global alignment. Let $S[i, j]$ denote the score of an optimal alignment between $x_1...x_m$ and $y_1...y_n$. By definition for the first row and first column, we have $S[0, 0] = 0, S[i, 0] = -\beta \times i$, and
Algorithm GLOBALALIGNMENTSCORE(X,Y)

\[ S[0, 0] \leftarrow 0 \]

for \( j \leftarrow 1 \) to \( n \) do
\[ S[0, j] \leftarrow -\beta \times j \]
end

for \( i \leftarrow 1 \) to \( m \) do
\[ S[i, 0] \leftarrow -\beta \times i \]
for \( j \leftarrow 1 \) to \( n \) do
\[ S[i, j] = \max \{ S[i-1, j] - \beta, S[i, j-1] - \beta, S[i-1, j-1] + \delta(x_i, y_j) \} \]
end

end

Output \( S[m, n] \)

Figure 3.5: Computation of the score of an optimal GA.

\[ S[0, j] = -\beta \times j. \] With these initialization, \( S[i, j] \) for \( i \in \{1, 2, ..., m\} \) and \( j \in \{1, 2, ..., n\} \) can be computed by the following formula:

\[ S[i, j] = \max\{ S[i-1, j] - \beta, S[i, j-1] - \beta, S[i-1, j-1] + \delta(x_i, y_j) \} \]

Figure 3.6 gives the pseudo-code for delivering an optimal global alignment, where an initial call GLOBALALIGNMENTOUTPUT(X,Y,S,i,j) is made to deliver an optimal global alignment. Let \((i, j)\) be the entry under consideration. If \(i = 1\) or \(j = 1\), we simply output all the insertion pairs or deletion pairs in these boundary conditions. Otherwise, consider the following three cases. If \(S[i, j] = S[i-1, j-1] + \delta(x_i, y_j)\), we make a diagonal move and output the substitution pair \((x_i y_j)\). If \(S[i, j] = S[i-1, j] - \beta\), then should make a vertical move and output a deletion pair \((x_i \phi)\). Otherwise, \(S[i, j] = S[i, j-1] - \beta\), then one needs to make an horizontal move and output an insertion pair \((\phi y_j)\).

3.2 Local Alignment: Smith-Waterman Algorithm

A modification of the Needleman-Wunsch algorithm is given in [11]. The goal of this new alignment is to find the highest scoring local match among the two sequences. In a lot of applications, a local alignment is preferred to the global alignment i.e., one takes a high-scoring local path which does not require terminate at the corners of the dynamic programming matrix, [4], [8] and [11]. Local alignment aims to find the best pair of sub-sequences from each sequence, such that the optimal alignment of these two regions
Algorithm GLOBALALIGNMENTOUTPUT(X,Y,S,i,j)

if $i = 1$ or $j = 1$ then
  if $i > 0$ then
    for $i' \leftarrow 1$ to $i$ do
      print $(x_{i'})$
    end
  end
  if $j > 0$ then
    for $j' \leftarrow 1$ to $j$ do
      print $(y_{j'})$
    end
  end
  return
end

if $S[i,j] = S[i-1,j-1] + \delta(x_i,y_j)$ then
  GLOBALALIGNMENTOUTPUT(X,Y,S,i-1,j-1)
  print $(x_i)$
else if $S[i,j] = S[i-1,j] - \beta$ then
  GLOBALALIGNMENTOUTPUT(X,Y,S,i-1,j)
  print $(y_j)$
else
  GLOBALALIGNMENTOUTPUT(X,Y,S,i,j-1)
  print $(y_j)$
end

Figure 3.6: Traceback procedure for delivering an optimal GA.

is the best possible. The sub-sequences may be part or all of the both sequences. If all of both are included then the local alignment is also global. The scoring matrix of local alignment is given by:

$$S[i,j] = \max\{0, S[i-1,j] - \beta, S[i,j-1] - \beta, S[i-1,j-1] + \delta(x_i,y_j)\}.$$  

The recurrence is similar to the global alignment except the first entry is zero i.e., if the scores of all the possible paths to the current position are all negative, they are reset to zero. The largest value of $S[i,j]$ is the score of the best local alignment between the sequences $X$ and $Y$. The local alignment starts from the highest-scoring position in the scoring matrix and traceback from this position up to a cell that scores zero. Denote this method as LA in this thesis. Now we explain the local alignment for our example, $X=$GCAGG and $Y=$AAAGTGG.
3.2.1 Initialization Step

In this step the row $S[i, 0]$ and the column $S[0, j]$ is set to zero i.e.,

$$S[0, 0] = 0, \quad S[i, 0] = 0, \quad S[0, j] = 0.$$ 

\[ \begin{array}{cccccc}
   & A & A & A & G & T & G & G \\
   G & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
   C & 0 &   &   &   &   &   &   \\
   A & 0 &   &   &   &   &   &   \\
   G & 0 &   &   &   &   &   &   \\
   G & 0 &   &   &   &   &   &   
\end{array} \]

**Figure 3.7: Initialization Step in LA.**

3.2.2 Matrix Filling Step

In this step we will find the maximum local alignment score by starting in the upper left hand corner in the matrix, and finding the maximal score $S[i, j]$ for every position in the matrix. For finding $S[i, j]$ for each $i, j$, one needs to know the score of the matrix positions of the above, diagonal and left i.e., $S[i, j - 1]$, $S[i - 1, j - 1]$ and $S[i - 1, j]$. The maximum score at position $(i, j)$ is defined as below:

$$S[i, j] = \max \left\{ 0, S[i - 1, j] - \beta, S[i, j - 1] - \beta, S[i - 1, j - 1] + \delta(x_i, y_j) \right\}.$$ 

With this information the score at position $S[1, 1]$ is $S[1, 1] = \max\{0, S[0, 0] - 2, S[1, 0] - 1, S[0, 1] - 1\} = \max\{0, -1, -1, -1\} = 0$. One can continue filling the rest of the matrix in a same way. Recall that if the value is not positive, we should put zero instead and no arrow is drawn back from this position. The matrix is shown in Figure 3.8.

3.2.3 Traceback Step

After the matrix filling step, the maximum local alignment score for these sequences is 11. We might have more than one maximum score in the local alignment. We start the traceback in the position with the highest value by looking where should move next according to the pointers. When we reach a cell with the zero value then we stop the
Chapter 3. Pairwise Sequence Alignments

We start from \((5, 6)\) where \(S[5, 6] = 11\). Since \(S[5, 6] = S[4, 5] + 4 = 7 + 4 = 11\), we make a diagonal move back to the entry \((4, 5)\), it corresponds to the substitution \((G \phi G)\) in the alignment. Since \(S[4, 5] = S[4, 4] - 1 = 8 - 1 = 7\), we make an horizontal move back to the entry \((4, 4)\), it corresponds to the insertion \((\phi T)\) in the alignment. Since \(S[4, 4] = S[3, 3] + 4 = 4 + 4 = 8\), should make a diagonal move back to the entry \((3, 3)\), it corresponds to the substitution \((G \phi G)\) in the alignment. Since \(S[3, 3] = S[2, 2] + 4 = 0 + 4 = 4\), we make a diagonal move back to the entry \((2, 2)\), it corresponds to the substitution \((A A)\) in the alignment. Since, we reach to \(S[2, 2]\) with the zero value the process should be stopped. Figure 3.9 shows the result of the local alignment.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>A</th>
<th>A</th>
<th>G</th>
<th>T</th>
<th>G</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>7</td>
<td>6</td>
<td>10</td>
</tr>
</tbody>
</table>

**Figure 3.8:** Matrix Filling Step in LA.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>A</th>
<th>A</th>
<th>G</th>
<th>T</th>
<th>G</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>7</td>
<td>6</td>
<td>10</td>
</tr>
</tbody>
</table>

**Figure 3.9:** Traceback Step in LA.

So the local alignment for these DNA sequences is:

\[
\begin{array}{ccccc}
A & G & \phi & G \\
A & G & T & G \\
+ & + & - & + \\
4 & 4 & 1 & 4 \\
\end{array}
\]

The above alignment includes three matches and one indel. So the score of an alignment is:
Algorithm LOCALALIGNMENTSCORE(X,Y)

\[
S[0, 0] \leftarrow 0 \\
Best \leftarrow 0 \\
Endi \leftarrow 0 \\
Endj \leftarrow 0 \\
\text{for } j \leftarrow 1 \text{ to } n \text{ do} \\
| \quad S[0, j] \leftarrow 0 \\
\text{end} \\
\text{for } i \leftarrow 1 \text{ to } m \text{ do} \\
| \quad S[i, 0] \leftarrow 0 \\
| \quad \text{for } j \leftarrow 1 \text{ to } n \text{ do} \\
| \quad | \quad S[i, j] = \max \left\{ 0, S[i - 1, j] - \beta, S[i, j - 1] - \beta, S[i - 1, j - 1] + \delta(x_i, y_j) \right\} \\
| \quad \text{end} \\
| \quad \text{if } S[i, j] > Best \text{ then} \\
| \quad | \quad Best \leftarrow S[i, j] \\
| \quad | \quad Endi \leftarrow i \\
| \quad | \quad Endj \leftarrow j \\
| \quad \text{end} \\
\text{end} \\
\text{Output } S[i, j], Endi, Endj
\]

**Figure 3.10:** Computation of the score of an optimal LA.

\[3 \times (+4) - 1 \times (+1) = 12 - 1 = 11\]

The total running time of the local alignment is \(O(mn)\) similar to the global alignment.

### 3.2.4 Local Alignment Algorithms

Figure 3.10 gives the pseudo-code for computing the score of an optimal local alignment. Let \(S[i, j]\) denote the score of an optimal alignment between \(x_1...x_m\) and \(y_1...y_n\). By definition for the first row and first column, we have \(S[0, 0] = 0, S[i, 0] = 0,\) and \(S[0, j] = 0\). We define \((Endi, Endj)\) as the indices of the position with higher score. At the beginning \((Endi, Endj)\) is equal to \((0, 0)\). We have the \(Best\) as the highest score of the local alignment, at the beginning it is equal to 0. With these initialization, \(S[i, j]\) for \(i \in \{1, 2, ..., m\}\) and \(j \in \{1, 2, ..., n\}\) can be computed by the following formula:

\[S[i, j] = \max \{0, S[i - 1, j] - \beta, S[i, j - 1] - \beta, S[i - 1, j - 1] + \delta(x_i, y_j)\}\]

If in each iteration, \(S[i, j]\) is higher than \(Best\), put \(S[i, j]\) in the \(Best\) and save the indices in the \((Endi, Endj)\).
Algorithm LOCALALIGNMENTOUTPUT(X,Y,S,i,j)

if \( S[i,j] = 0 \) then
    return
end
if \( S[i,j] = S[i-1,j-1] + \delta(x_i,y_j) \) then
    LOCALALIGNMENTOUTPUT(X,Y,S,i-1,j-1)
    print \( (x_i,y_j) \)
end
if \( S[i,j] = S[i-1,j] - \beta \) then
    LOCALALIGNMENTOUTPUT(X,Y,S,i-1,j)
    print \( (x_i,\varnothing) \)
else
    LOCALALIGNMENTOUTPUT(X,Y,S,i,j-1)
    print \( (\varnothing,y_j) \)
end

---

Figure 3.11: Traceback procedure for delivering an optimal LA.

Figure 3.11 gives the pseudo-code for delivering an optimal local alignment, where an initial call LOCALALIGNMENTOUTPUT(X,Y,S,i,j) is made to deliver an optimal local alignment. Let \((i,j)\) be the entry under consideration. We trace back the dynamic programming matrix from the maximum-score entry \((\text{End}_i,\text{End}_j)\) recursively according to the following rules. If \( S[i,j] = 0 \), we have reached the beginning of the optimal local alignment. Otherwise, we have these three cases. If \( S[i,j] = S[i-1,j-1] + \delta(x_i,y_j) \), we make a diagonal move and output a substitution pair \( (x_i,y_j) \). If \( S[i,j] = S[i-1,j] - \beta \), then we make a vertical move and output a deletion pair \( (x_i,\varnothing) \). Otherwise, \( S[i,j] = S[i,j-1] - \beta \), then we make a horizontal move and output an insertion pair \( (\varnothing,y_j) \).

### 3.3 Semi-Global Alignment

We discussed two different sequence alignments: global alignment and local alignment. Now we introduce another kind of alignment which is called Semi-Global Alignment. We denote this method as SGA in the rest of this thesis. Semi-Global Alignment is similar to the global alignment, the differences between them is the way for scoring alignments and the traceback step. Semi-Global Alignment does not penalize indels at the beginning or at the ending of the alignment but in global alignment there is no difference between indels and we penalize all of them in the alignment. In other words semi-global alignment does not give any cost to indels that appear before the first character and after the last character. The traceback in the semi-global alignment starts at the maximum of the last column.
(S[i, n]) or last row (S[m, j]) and ends at any cell in the first row or first column, [13]. Let us explain our example and apply the semi-global alignment for it.

3.3.1 Initialization Step

In this step each row $S[i, 0]$ and each column $S[0, j]$ is set to zero

$$S[0, 0] = 0, \quad S[i, 0] = 0, \quad S[0, j] = 0.$$ 

Using this scoring schema, the initialization step results is given in Figure 3.12.

![Figure 3.12: Initialization Step in SGA.](image)

3.3.2 Matrix Filling Step

To find the maximum semi-global alignment score, start from the upper left hand corner of the matrix and finding the maximal score $S[i, j]$ for each position, see Figure 3.13 in the next page.

Figure 3.13 presents three possible ways to enter to the grid point $(i, j)$, and one should choose the maximum of their path weights. The weight of maximum-scoring path entering $(i, j)$ from $(i - 1, j)$ vertically is the weight of maximum-scoring path entering $(i - 1, j)$ plus the weight of edge $(i - 1, j) \rightarrow (i, j)$. Therefore the weight of maximum-scoring path entering $(i, j)$ with a deletion gap symbol at the end is $S[i - 1, j] - \beta$.

Similarly weight of maximum-scoring path entering $(i, j)$ from $(i, j - 1)$ horizontally is the weight of maximum-scoring path entering $(i, j - 1)$ plus the weight of $(i, j - 1) \rightarrow (i, j)$. Therefore the weight of maximum-scoring path entering $(i, j)$ with an insertion gap symbol at the end is $S[i, j - 1] - \beta$.

Finally the weight of maximum-scoring path entering $(i, j)$ from $(i - 1, j - 1)$ diagonally
Chapter 3. Pairwise Sequence Alignments

Figure 3.13: The ways entering to the point \((i,j)\) in SGA.

is \(S[i-1, j-1] + \delta(x_i, y_j)\).

We simply take the maximum value of these three choices to compute \(S[i,j]\).

\[
S[i,j] = \max \{ S[i-1,j] - \beta, S[i,j-1] - \beta, S[i-1,j-1] + \delta(x_i, y_j) \}.
\]

Using the above information, the score at position \(S[1,1]\) in the matrix can be calculated as:

\[
S[1,1] = \max \{ S[0,0] - 2, S[1,0] - 1, S[0,1] - 1 \} = \max \{-2, -1, -1\} = -1.
\]

The value of -1 is then placed in position \(S[1,1]\). Now we proceed to \(S[1,2] = \max \{ S[0,1] - 2, S[1,1] - 1, S[0,2] - 1 \} = \max \{-2, -2, -1\} = -1\). The procedure should be continued until filling the rest of the matrix. The resulting matrix is given in Figure 3.14.

**Figure 3.14:** Matrix Filling Step in SGA
3.3.3 Traceback Step

After filling the matrix, the maximum of semi-global alignment of these sequences is 9. The traceback begins in the position $S[i, n]$ or $S[m, j]$ and it will be end in any cell in the first row or first column. We can have more than one maximum, for example in the Figure 3.14, we have two maximum ($S[5, 7]$ and $S[5, 6]$). We start from $S[5, 7]$ or $S[5, 6]$ and continue the path until reaching one of the situations shown in the Figure 3.15.

At the beginning, the possible predecessor is diagonal. In the upper path, $S[5, 7] = S[4, 6] + 4 = 5 + 4 = 9$, we make a diagonal move back to the entry $(4, 6)$, it corresponds to the substitution $(G)$ in the alignment. Since $S[4, 6] = S[3, 5] + 4 = 1 + 4 = 5$, we make a diagonal move back to the entry $(3, 5)$, it corresponds to the substitution $(G)$ in the alignment. Since $S[3, 5] = S[2, 4] - 2 = 3 - 2 = 1$, we make a diagonal move back to the entry $(2, 4)$, it corresponds to the substitution $(A)$ in the alignment. Since $S[2, 4] = S[1, 4] - 1 = 4 - 1 = 3$, we make a vertical move back to the entry $(1, 4)$, it corresponds to a deletion $(C)$ in the alignment. Since $S[1, 4] = S[0, 3] + 4 = 0 + 4 = 4$, we make a diagonal move back to the entry $(0, 3)$, it corresponds to the substitution $(G)$ in the alignment.

In the lower path, $S[5, 6] = S[4, 5] + 4 = 5 + 4 = 9$, we make a diagonal move back to the entry $(4, 5)$, it corresponds to the substitution $(G)$ in the alignment. Since $S[4, 5] = S[4, 4] - 1 = 6 - 1 = 5$, we make an horizontal move back to the entry $(4, 4)$, it corresponds to an insertion $(T)$ in the alignment. Since $S[4, 4] = S[3, 3] + 4 = 2 + 4 = 6$, we make a diagonal move back to the entry $(3, 3)$, it corresponds to the substitution $(G)$ in the alignment. Since $S[3, 3] = S[2, 2] + 4 = -2 + 4 = 2$, we make a diagonal move back to the entry $(2, 2)$, it corresponds to the substitution $(A)$ in the alignment. Since $S[2, 2] = S[2, 1] - 1 = -1 - 1 = -2$, we make an horizontal move back to the entry $(2, 1)$, it corresponds to an insertion $(A)$ in the alignment. Since $S[2, 1] = S[2, 0] - 1 = 0 - 1 = -1$, we make an horizontal move back to the entry $(2, 0)$, it corresponds to an insertion $(A)$ in the alignment. Since, we reach to $S[0, 3]$ in the first row the process should be stopped.

The resulting semi-global alignments can be any of them:

1) Upper path:

$$
\begin{array}{cccccc}
G & C & A & G & G \\
G & \phi & T & G & G \\
+ & - & - & + & + \\
4 & 1 & 2 & 4 & 4 \\
\end{array}
$$
The above alignment includes three matches, one mismatch and one indel. Hence, the total score of the alignment is:

\[ 3 \times (+4) + 1 \times (-2) - 1 \times (+1) = 12 - 2 - 1 = 9 \]

2) Lower Path:

\[
\begin{array}{cccccc}
\phi & \phi & A & G & \phi & G \\
A & A & A & G & T & G \\
- & - & + & + & - & + \\
1 & 1 & 4 & 4 & 1 & 4 \\
\end{array}
\]

The above alignment includes three matches and three indels. Hence, the total score of the alignment is:

\[ 3 \times (+4) - 3 \times (+1) = 12 - 3 = 9 \]

The total running time of the semi-global alignment is \( O(mn) \) similar to the global alignment.

### 3.3.4 Semi-Global Algorithms

Figure 3.16 gives the pseudo-code for computing the score of an optimal semi-global alignment. Let \( S[i, j] \) denote the score of an optimal alignment between \( x_1...x_m \) and \( y_1...y_n \). By definition for the first row and first column, we have \( S[0, 0] = 0, S[i, 0] = 0, \) and \( S[0, j] = 0 \). We define \((End_i, End_j)\) as the indices of the position with higher score. At the beginning \((End_i, End_j)\) is equal to \((0, 0)\). We have the Best as the highest score of the semi-global alignment, at the beginning it is equal to 0. With these initialization, \( S[i, j] \) for \( i \in \{1, 2, ..., m\} \) and \( j \in \{1, 2, ..., n\} \) can be computed by the following formula:

\[
S[i, j] = \max \{ S[i - 1, j] - \beta, S[i, j - 1] - \beta, S[i - 1, j - 1] + \delta(x_i, y_j) \}.
\]
Algorithm SEMIGLOBALALIGNMENTSCORE(X,Y)

\[ S[0,0] \leftarrow 0 \]
Best \leftarrow 0; Endi \leftarrow 0; Endj \leftarrow 0

for \( j \leftarrow 1 \) to \( n \) do
| \( S[0,j] \leftarrow 0 \)
end

for \( i \leftarrow 1 \) to \( m \) do
| \( S[i,0] \leftarrow 0 \)
end

for \( i \leftarrow 2 \) to \( m - 1 \) do
| for \( j \leftarrow 2 \) to \( n - 1 \) do
| \( S[i,j] = \max \begin{cases} 
S[i-1,j] - \beta \\
S[i,j-1] - \beta \\
S[i-1,j-1] + \delta(x_i,y_j) 
\end{cases} \)
| end
| if \( S[i,j] > \text{Best} \) then
| \( \text{Best} \leftarrow S[i,j]; \text{Endi} \leftarrow i; \text{Endj} \leftarrow j \)
| end
end

end

for \( j \leftarrow 2 \) to \( n \) do
| \( S[i,j] = \max \begin{cases} 
S[i-1,j] - \beta \\
S[i,j-1] - \beta \\
S[i-1,j-1] + \delta(x_i,y_j) 
\end{cases} \)
| \( \text{if } S[i,j] > \text{Best} \text{ then} \)
| \( \text{Best} \leftarrow S[i,j]; \text{Endi} \leftarrow i; \text{Endj} \leftarrow j \)
| \( \text{end} \)
end

Output \( S[m,n]; \text{Endi}; \text{Endj} \)

Figure 3.16: Computation of the score of an optimal SGA.
We want to find the maximum score in the last column or last row. For each iteration in the last row, if \( S[m,j] \) \((j \in \{2, ..., n\})\), is higher than \( \text{Best} \), put \( S[m,j] \) in the \( \text{Best} \) and save the indices in the \((\text{End}_i, \text{End}_j)\). For each iteration in the last column, if \( S[i,n] \) \((i \in \{2, ..., m\})\), is higher than \( \text{Best} \), put \( S[i,n] \) in the \( \text{Best} \) and save the indices in the \((\text{End}_i, \text{End}_j)\).

Figure 3.17 gives the pseudo-code for delivering an optimal semi-global alignment, where an initial call \( \text{SEMIGLOBALALIGNMENTOUTPUT}(X,Y,S,i,j) \) is made to deliver an optimal semi-global alignment. Let \((i,j)\) be the entry under consideration. We trace back the dynamic programming matrix from the maximum-score entry \((\text{End}_i, \text{End}_j)\) from the last row or last column recursively according to the following rules. If \( S[i,j] = 0 \), we have reached the beginning of the optimal semi-global alignment. Otherwise, we have these three cases. if \( S[i,j] = S[i-1,j-1] + \delta(x_i,y_j) \), we make a diagonal move and output a substitution pair \((x_i,y_j)\). If \( S[i,j] = S[i-1,j] - \beta \), then we make a vertical move and output a deletion pair \((x_i,)\). Otherwise, \( S[i,j] = S[i,j-1] - \beta \), then we make an horizontal move and output an insertion pair \((,y_j)\).

In this chapter, we introduced three existing sequence alignments: global alignment, local alignment, semi-global alignment. Global alignment attempts to match two sequences to each other from end to end. Local alignment that finds the segments of two sequences that match well, just those parts that appear to have a good similarity. Semi-global alignment is similar to the global alignment. The differences between them are the way for the scoring alignments and choosing the maximum of last column and last row to start the traceback. Semi-global alignment does not penalize indels at the
beginning or at the ending of the alignment but in global alignment there is no difference between indels and we penalize all of them in the alignment.

The discussion of comparison of sequences in terms of GA and LA can be found in [4]. If the statistical approach is of concern, [6] and [12] are the right ones.
Chapter 4

New Sequence Alignments

In this section two new sequence alignments are proposed. These two methods extend the global alignment by using a more appropriate cost of the indel cells rather than the constant cost.

4.1 Weighted Alignment

In the global alignment, the movements to the right or down are performed using the equal cost, but one may adopt the appropriate cost according to the movement or position. We refer to it as weighted alignment, denoted as WA hereafter. Different approaches may be considered in the weighted alignment, here the idea is to choose the weights for the indel such that the flow of movements become closer to the diagonal. Under the homogeneous sequences, it is expected that the same nucleotides are located on the diagonal of the alignment matrix. Therefore, any of them that are closer to the diagonal should have less weight than the other. For example consider Figure 4.1.

![Figure 4.1](image-url)
Chapter 4. New Sequence Alignments

In the above part of the diagonal, for the cells that are far from the diagonal or the center of the table, the horizontal movement should have lower weight than the vertical movement. By horizontal movement, we can reach the diagonal quickly. In the lower part of the diagonal, for the cells that are far from the diagonal or the center of the table, the vertical movement should have lower weight than the horizontal movement because by vertical movement we can reach the diagonal. Let us denote $w$ as the weight, the similarity is therefore defined by:

$$S[i, j] = \max\{S[i - 1, j] - w_{i-1,j} \beta, S[i, j - 1] - w_{i,j-1} \beta, S[i - 1, j - 1] + w_{i-1,j-1} \delta(x_i, y_j)\}.$$  \hspace{1cm} (4.1)

In order to move vertically, the weight of the vertical line should be lower than the weight of the horizontal line, $w_{i-1,j} < w_{i,j-1}$, and if the aim is the horizontal movement then the weight of the horizontal line should be lower than the weight of the vertical line $w_{i,j-1} < w_{i-1,j}$. For the cells that are near the diagonal or center of the table, the weights of horizontal and vertical are very close to each other. It means that the weights are approximately the same $w_{i-1,j} \approx w_{i,j-1}$. The choice of weight can be determined in terms of the position of the cell and its direction, for instance, $w_{i-1,j} = 1/(m - i + 1)$ and $w_{i,j-1} = 1/(n - j + 1)$. By substitution this in (4.1), we have:

$$S[i, j] = \max\{S[i - 1, j] - \frac{1}{m - i + 1} \beta, S[i, j - 1] - \frac{1}{n - j + 1} \beta, S[i - 1, j - 1] + \varrho \delta(x_i, y_j)\},$$  \hspace{1cm} (4.2)

where

$$\varrho = \max\left\{\frac{1}{m - i + 1}, \frac{1}{n - j + 1}\right\}.$$  

By choosing such a weight, actually we may force the movement closer to the diagonal. In order to adjust the first column and the first row we consider:

$$S[0, 0] = 0,$$

$$S[0, j] = S[0, j - 1] - \frac{1}{n - j + 1} \beta,$$

$$S[i, 0] = S[i - 1, 0] - \frac{1}{m - i + 1} \beta.$$  \hspace{1cm} (4.3)
### 4.1.1 Initialization Step

In this step each row $S[i,0]$ and each column $S[0,j]$ is set by using formula (4.3). Using this scoring schema, the initialization step outcomes are shown in Figure 4.2.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>A</th>
<th>A</th>
<th>G</th>
<th>T</th>
<th>G</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>0.00</td>
<td>-0.14</td>
<td>-0.30</td>
<td>-0.50</td>
<td>-0.75</td>
<td>-1.09</td>
<td>-1.59</td>
</tr>
<tr>
<td>C</td>
<td>-0.20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-0.45</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>-0.78</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>-1.28</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>-2.28</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Figure 4.2: Initialization step in WA.](image)

### 4.1.2 Matrix Filling Step

Using the formula (4.2) we could find the maximum value for each cell according to its position. It can be therefore used to fill out the matrix and to find the score for each cell. The matrix is shown in Figure 4.3.

![Figure 4.3: Matrix Filling step in WA.](image)

### 4.1.3 Traceback Step

After the matrix filling step, the maximum of weighted alignment for two sequences is 5.99. The traceback begins in position $S[m,n]$ and it will end in the position $S[0,0]$. At
each cell, we seek to determine where is the next move, according to the pointers. One could follow the path until reaching one of the following situations presented in Figure 4.4.

With the traceback step, we start from $S[5, 7]$ and it will end in $S[0, 0]$. Initially, the possible predecessor is a diagonal movement. Using the formula (4.2), in the upper path, $S[5, 7] = S[4, 6] + 4 = 1.99 + 4 = 5.99$, we make a diagonal move back to the entry $(4, 6)$, which corresponds to the substitution $(G_i)$ in the alignment. Since $S[4, 6] = S[3, 5] + \frac{1}{2} \times 4 = -0.009 + 2 = 1.99$ ($\varrho = \frac{1}{2}$), we make a diagonal move back to the entry $(3, 5)$, which corresponds to the substitution $(G_i)$ in the alignment. Since $S[3, 5] = S[3, 4] - \frac{1}{3} \times 1 = 0.31 - \frac{1}{3} = -0.009$, we make an horizontal move back to the entry $(3, 4)$, which corresponds to an insertion ($\varphi T_i$) in the alignment. The procedure continues until reaching $S[0, 0]$.

![Figure 4.4: Traceback Step in WA.](image)

The resulting weighted alignments can be any of the following:

1) Upper path:

$$
\begin{align*}
&\phi & & \phi & & G & & C & & A & & \phi & & \phi & & G & & G \\
& A & & A & & \phi & & \phi & & A & & G & & T & & G & & G \\
& \frac{1}{7} & & \frac{1}{5} & & \frac{1}{5} & & \frac{1}{4} & & \frac{1}{3} & & \frac{1}{4} & & \frac{1}{3} & & 2 & & 4
\end{align*}
$$

The above alignment includes three matches and six indels (two insertions and four deletions). The score for match and mismatch is $\{\max\{\frac{1}{m+1}, \frac{1}{n+1}\} \delta(x_i, y_j)\}$. The gap penalty for an insertion (horizontal move) is $\{-\frac{1}{n+1} \beta\}$ and for a deletion (vertical move) is $\{-\frac{1}{m+1} \beta\}$. Therefore the score of an alignment is:

$$
\begin{align*}
&-\frac{1}{7} - \frac{1}{5} - \frac{1}{5} + \frac{1}{4} + \frac{1}{3} \times (+4) - \frac{1}{r-4+1} - \frac{1}{r-6+1} + \frac{1}{2} \times (+4) + 1 \times 4 = \\
&\frac{1}{7} - \frac{1}{6} - \frac{1}{4} + \frac{1}{3} - \frac{1}{2} + 2 + 4 = 5.99
\end{align*}
$$
Algorithm WEIGHTEDALIGNMENTSCORE(X,Y)

\[ S[0,0] \leftarrow 0 \]
for \( j \leftarrow 1 \) to \( n \) do
\[ S[0,j] \leftarrow S[0,j-1] - \frac{1}{n-j+1} \beta \]
end
for \( i \leftarrow 1 \) to \( m \) do
\[ S[i,0] \leftarrow S[i-1,0] - \frac{1}{m-i+1} \beta \]
for \( j \leftarrow 1 \) to \( n \) do
\[ \rho \leftarrow \max\{\frac{1}{m-i+1}, \frac{1}{n-j+1}\} \]
\[ S[i,j] = \max\{S[i-1,j] - \frac{1}{m-i+1} \beta, S[i,j-1] - \frac{1}{n-j+1} \beta, S[i-1,j-1] + \rho \delta(x_i, y_j)\} \]
end
end
Output \( S[m,n] \)

Figure 4.5: Computation of the score of an optimal WA.

2) Lower path:

\[
\begin{align*}
G & \quad C & \phi & \phi & A & \phi & \phi & G & G \\
\phi & \quad \phi & A & A & A & G & T & G & G \\
- & - & - & - & + & - & + & + \\
\frac{1}{5} & \quad \frac{1}{4} & \frac{1}{7} & \frac{1}{6} & \frac{4}{3} & \frac{1}{4} & \frac{1}{3} & 2 & 4 \\
\end{align*}
\]

The above alignment includes three matches and six indels (four insertions and two deletions). Therefore the score of an alignment is:
\[
-\frac{1}{5} - \frac{1}{4} - \frac{1}{7} - \frac{1}{6} + \frac{4}{3} - \frac{1}{4} - \frac{1}{3} + 2 + 4 = \frac{1}{2} \times (+4) - \frac{1}{7} = 5.99
\]

The total running time of the weighted alignment is \( O(mn) \), similar to the global alignment.

4.1.4 Weighted Alignment Algorithms

Figure 4.5 gives the pseudo-code for computing the score of an optimal weighted alignment. Let \( S[i,j] \) denote the score of an optimal alignment between \( x_1...x_m \) and \( y_1...y_n \).

By definition for the first row and first column, we have \( S[0,0] = 0 \), \( S[i,0] = S[i-1,0] - \frac{1}{m-i+1} \beta \), and \( S[0,j] = S[0,j-1] - \frac{1}{n-j+1} \beta \). With these initializations, \( S[i,j] \) for
Algorithm: WEIGHTEDALIGNMENTOUTPUT(X,Y,S,i,j)

if $i = 1$ or $j = 1$ then
  if $i > 0$ then
    for $i' \leftarrow 1$ to $i$ do
      print $(x_{i'})$
    end
  end
  if $j > 0$ then
    for $j' \leftarrow 1$ to $j$ do
      print $(y_{j'})$
    end
  end
else
  $S[i,j] = S[i-1,j-1] + \max\{1_{m-i+1}, 1_{n-j+1}\} \cdot \delta(x_i, y_j)$ then
    WEIGHTEDALIGNMENTOUTPUT(X,Y,S,i-1,j-1)
    print $(x_i, y_j)$
  else if $S[i,j] = S[i-1,j] - \frac{1}{m-i+1} \cdot \beta$ then
    WEIGHTEDALIGNMENTOUTPUT(X,Y,S,i-1,j)
    print $(x_i, y_j)$
  else
    WEIGHTEDALIGNMENTOUTPUT(X,Y,S,i,j-1)
    print $(y_{j'}, x_{i'})$
end

Figure 4.6: Traceback procedure for delivering an optimal WA.

$i \in \{1, 2, ..., m\}$ and $j \in \{1, 2, ..., n\}$ can be computed using the following formula:

$$S[i,j] = \max\{S[i-1,j] - \frac{1}{m-i+1} \cdot \beta, S[i,j-1] - \frac{1}{n-j+1} \cdot \beta, S[i-1,j-1] + \varrho \cdot \delta(x_i, y_j)\}.$$  \hspace{1cm} (4.4)

where

$$\varrho = \max\{\frac{1}{m-i+1}, \frac{1}{n-j+1}\}.$$  

Figure 4.6 gives the pseudo-code for delivering an optimal weighted alignment, where an initial call WEIGHTEDALIGNMENTOUTPUT(X,Y,S,i,j) is made to deliver an optimal weighted alignment. Let $(i,j)$ be the entry under consideration. If $i = 1$ or $j = 1$, we simply output all the insertion pairs or deletion pairs in these boundary conditions. Otherwise, consider the following three cases. If $S[i,j] = S[i-1,j-1] + \max\{\frac{1}{m-i+1}, \frac{1}{n-j+1}\} \cdot \delta(x_i, y_j)$, we make a diagonal move and output the substitution pair...
If \( S[i, j] = S[i - 1, j] - \frac{1}{m-i+1} \beta \), we make a vertical move and output a deletion pair \((^x_i, y_j)\). Otherwise, \( S[i, j] = S[i, j - 1] - \frac{1}{n-j+1} \beta \) and we should execute an horizontal move and output an insertion pair \((^x_i, y_j)\).

### 4.2 Diagonal Alignment

In this approach, we draw the diagonals of the table. Since there are sequences with different lengths, one diagonal starts from the leftmost cell of the first row, and the other one starts from the rightmost cell of the last row. The weight should be calculated according to the location of the cells in relation to the diagonal. Consider Figure 4.7 that includes two diagonals.

![Figure 4.7](attachment:image.png)

We assign the following score to the initialized cells.

\[
S[0, 0] = 0, \quad S[i, 0] = -\beta \times i, \quad S[0, j] = -\beta \times j,
\]

and the other ones can be calculated using the following procedures:

i. Cells in the lower triangle (including the lower diagonal): Consider the following assignment, it shows that the vertical movement is higher than the horizontal movement, see Figure 4.8.

\[
S[i, j] = \max \{ S[i - 1, j] - \beta, S[i, j - 1] - \beta, S[i - 1, j - 1] + \delta(x_i, y_j) \}.
\]

ii. Cells between the diagonals: The following statement shows that there is no priority.

\[
S[i, j] = \max \{ S[i - 1, j] - \beta, S[i, j - 1] - \beta, S[i - 1, j - 1] + \delta(x_i, y_j) \}.
\]

iii. Cells in the upper triangle (including the upper diagonal): Consider the following assignment, it shows that the horizontal movement is higher than the vertical movement, see Figure 4.9.
Figure 4.8: Direct for the lower triangle using DA.

\[ S[i,j] = \max\{S[i-1,j] - \beta, S[i,j-1], S[i-1,j-1] + \delta(x_i, y_j)\}. \]

We denote this method as a DA in rest of this thesis. Let us explain these three steps using our example to help understand it better.

### 4.2.1 Initialization Step

In this step each row \( S[i,0] \) is set to \( -\beta \times i \) and each column \( S[0,j] \) is set to \( -\beta \times j \). Recall that \( \beta \) is a gap penalty, so we have:

\[ S[0,0] = 0, \ S[i,0] = -\beta \times i, \ S[0,j] = -\beta \times j. \]
Using this scoring schema, the initialization step result is shown in Figure 4.10.

### 4.2.2 Matrix Filling Step

Using the explained procedures, the score at position $S[1,1]$, that is a cell in the lower triangle, can be calculated as:

$$S[1,1] = \max\{S[0,1], S[1,0] - 1, S[0,0] - 2\} = \max\{-1, -2, -2\} = -1.$$  

So the value of $-1$ is placed in position $S[1,1]$. $S[1,2]$ is a cell between the diagonals, using the proposed procedure:

$$S[1,2] = \max\{S[0,2] - 1, S[1,1] - 1, S[0,1] - 2\} = \max\{-3, -2, -3\} = -2,$$

so the value of $-2$ is placed in position $S[1,2]$. The procedure continues to fill the rest of the matrix. The resulting matrix is given in the Figure 4.11.

### 4.2.3 Traceback Step

After the matrix filling step, the maximum diagonal alignment for two sequences is 9. The traceback step will gain the actual alignments that results in the maximum
score. The traceback begins in position $S[m, n]$ and it will end in the position $S[0, 0]$. At each cell we seek to determine where is the next movement according to the pointers. Initially, the possible predecessors are diagonal or horizontal. In the upper path, $S[5, 7] = S[4, 6] + 4 = 5 + 4 = 9$ (upper triangle), we make a diagonal move back to the entry $(4, 6)$, this corresponds to the substitution $(\text{G}\text{G})$ in the alignment. Since $S[4, 6] = S[3, 5] + 4 = 1 + 4 = 5$ (upper triangle), we make a diagonal move back to the entry $(3, 5)$, which corresponds to the substitution $(\text{G}\text{G})$ in the alignment. Since $S[3, 5] = S[3, 4] = 1$ (upper triangle), we make an horizontal move back to the entry $(3, 4)$, and this corresponds to an insertion $(\phi)$ in the alignment. Since $S[3, 4] = S[3, 3] - 1 = 2 - 1 = 1$ (between diagonals), we make an horizontal move back to the entry $(3, 3)$, it corresponds to an insertion $(\phi)$ in the alignment. We continue the path until reaching $S[0, 0]$.

In the lower path, $S[5, 7] = S[5, 6] = 9$ (upper triangle), we make an horizontal move back to the entry $(5, 6)$, it corresponds to an insertion $(\phi)$ in the alignment. Since $S[5, 6] = S[4, 5] + 4 = 5 + 4 = 9$ (between diagonals), we make a diagonal move back to the entry $(4, 5)$, which corresponds to the substitution $(\text{G}\text{G})$ in the alignment. Since $S[4, 5] = S[4, 4] - 1 = 6 - 1 = 5$ (between diagonals), we make an horizontal move back to the entry $(4, 4)$, and this corresponds to an insertion $(\phi)$ in the alignment. Since $S[4, 4] = S[3, 3] + 4 = 2 + 4 = 6$ (lower triangle), we make a diagonal move back to the entry $(3, 3)$, which corresponds to the substitution $(\text{G}\text{G})$ in the alignment. We continue the path until we reach $S[0, 0]$.

By continuing the procedure, we arrive at the situations presented in Figure 4.12.

![Figure 4.12: Traceback Step in DA.](image)

The resulting diagonal alignments can be any of the following:
1) Upper path:

\[
\begin{align*}
\phi & \quad G & \quad \phi & \quad C & \quad A & \quad \phi & \quad \phi & \quad G & \quad G \\
A & \quad \phi & \quad A & \quad \phi & \quad A & \quad G & \quad T & \quad G & \quad G \\
- & + & - & + & - & + & + & + & + \\
1 & 0 & 1 & 0 & 4 & 1 & 0 & 4 & 4
\end{align*}
\]

To account the score, we should consider which movement is chosen, if it is in the triangle below the diagonal, the vertical movement has no penalty and if it is in the triangle above the diagonal, the horizontal movement has no penalty. The above alignment includes three matches and six indels (1st, 3rd and 6th columns are between the diagonals, 2nd and 4th columns are in the lower triangle and 7th column is in the upper triangle). Hence, the score of an alignment is:

\[-1 + 0 - 1 + 0 + 4 - 1 + 0 + 4 + 4 = 9\]

2) Lower path:

\[
\begin{align*}
\phi & \quad G & \quad C & \quad A & \quad \phi & \quad G & \quad \phi & \quad G & \quad \phi \\
A & \quad \phi & \quad \phi & \quad A & \quad A & \quad G & \quad T & \quad G & \quad G \\
- & + & + & + & - & + & - & + & + \\
1 & 0 & 0 & 4 & 1 & 4 & 1 & 4 & 0
\end{align*}
\]

The above alignment includes three matches and six indels (1st and 7th columns are between diagonals, 2nd, 3rd and 5th columns are in the lower triangle and 9th column is in the upper triangle). Hence, the score of an alignment is:

\[-1 + 0 + 0 + 4 - 1 + 4 - 1 + 4 + 0 = 9\]

The total running time of the diagonal alignment is \(O(mn)\), which is similar to the global alignment.

### 4.2.4 Diagonal Alignment Algorithms

Figure 4.13 gives the pseudo-code for computing the score of an optimal diagonal alignment. Let \(S[i, j]\) denote the score of an optimal alignment between \(x_1...x_m\) and \(y_1...y_n\).

By definition for the first row and first column, we have \(S[0, 0] = 0\), \(S[i, 0] = -\beta \times i\), and \(S[0, j] = -\beta \times j\). With these initializations, \(S[i, j]\) for \(i < j\) and \(j < i + (n - m)\) can be computed by the following formula:
Algorithm DIAGONALALIGNMENTSCORE(X,Y)

\[
S[0, 0] \leftarrow 0 \quad \text{for } j \leftarrow 1 \text{ to } n \text{ do} \\
\quad S[0,j] \leftarrow -\beta \times j \\
\text{end} \\
\text{for } i \leftarrow 1 \text{ to } m \text{ do} \\
\quad S[i, 0] \leftarrow -\beta \times i \\
\quad \text{for } j \leftarrow 1 \text{ to } n \text{ do} \\
\quad \quad \text{if } i < j \text{ and } j < i + (n - m) \text{ then} \\
\quad \quad \quad S[i,j] = \max \{ S[i-1,j] - \beta, S[i,j-1] - \beta, S[i-1,j-1] + \delta(x_i, y_j) \} \\
\quad \quad \text{else if } j \geq i + (n - m) \text{ then} \\
\quad \quad \quad S[i,j] = \max \{ S[i-1,j] - \beta, S[i,j-1] \} \\
\quad \quad \quad \text{else} \\
\quad \quad \quad \quad S[i,j] = \max \{ S[i-1,j], S[i,j-1] - \beta, S[i-1,j-1] + \delta(x_i, y_j) \} \\
\quad \text{end} \\
\text{end} \\
\text{end} \\
\text{Output } S[m, n]
\]

Figure 4.13: Computation of the score of an optimal DA.

\[
S[i,j] = \max \{ S[i-1,j] - \beta, S[i,j-1] - \beta, S[i-1,j-1] + \delta(x_i, y_j) \}.
\]

For \( j \geq i + (n - m) \), \( S[i,j] \) can be computed by the following formula:

\[
S[i,j] = \max \{ S[i-1,j] - \beta, S[i,j-1], S[i-1,j-1] + \delta(x_i, y_j) \}.
\]

Otherwise:

\[
S[i,j] = \max \{ S[i-1,j], S[i,j-1] - \beta, S[i-1,j-1] + \delta(x_i, y_j) \}.
\]

Figure 4.14-4.15 give the pseudo-code for delivering an optimal diagonal alignment, where an initial call DIAGONALALIGNMENTOUTPUT(X,Y,S,i,j) is made to deliver an optimal diagonal alignment. Let \((i, j)\) be the entry under consideration. If \( i = 1 \) or \( j = 1 \), we simply output all the insertion pairs or deletion pairs in these boundary conditions. Otherwise, consider the following three cases. For \( i < j \) and \( j < i + (n + m) \) (between diagonals), if \( S[i,j] = S[i-1,j-1] + \delta(x_i, y_j) \), we make a diagonal move and output the substitution pair \((x_i, y_j)\). If \( S[i,j] = S[i-1,j] - \beta \), then we make a vertical
Figure 4.14: Traceback procedure for delivering an optimal DA.

move and output a deletion pair \((x_i \phi)\). Otherwise, \(S[i, j] = S[i, j - 1] - \beta\) and should make an horizontal move and output an insertion pair \((\phi y_j)\).

For \(j \leq i + (n - m)\) (upper triangle), if \(S[i, j] = S[i - 1, j - 1] + \delta(x_i, y_j)\) then we make a diagonal move and output the substitution pair \((x_i y_j)\). Otherwise, \(S[i, j] = S[i - 1, j] - \beta\), we make an horizontal move and output an insertion pair \((\phi y_j)\).

In the lower triangle, if \(S[i, j] = S[i - 1, j]\), we make a vertical move and output a deletion pair \((x_i \phi)\). If \(S[i, j] = S[i, j - 1] - \beta\), then we make an horizontal move and output an insertion pair \((\phi y_j)\). Otherwise, \(S[i, j] = S[i - 1, j - 1] + \delta(x_i, y_j)\), we make a diagonal move and output the substitution pair \((x_i y_j)\).

In this chapter, we have proposed to improvement of the global alignment by having dynamic weight. We proposed two new alignments: weighted alignment and diagonal alignment. In the weighted alignment, we draw the diagonal of the table and divided the table in two parts, upper part and lower part. Here the idea is to choose the weights
Continue of Algorithm \textsc{Diagonalalignmentoutput}(X,Y,S,i,j)

\textbf{if} \( j \geq i + (n - m) \) \textbf{then}

\begin{itemize}
  \item \textbf{if} \( S[i,j] = S[i-1,j] - \beta \) \textbf{then}
    \begin{itemize}
      \item \textsc{Diagonalalignmentoutput}(X,Y,S,i-1,j)
      \item print \((x_i)\)
    \end{itemize}
  \item \textbf{else if} \( S[i,j] = S[i-j] \) \textbf{then}
    \begin{itemize}
      \item \textsc{Diagonalalignmentoutput}(X,Y,S,i,j-1)
      \item print \((\phi_y)\)
    \end{itemize}
  \item \textbf{else}
    \begin{itemize}
      \item \textsc{Diagonalalignmentoutput}(X,Y,S,i-1,j-1)
      \item print \((x_i)\)
    \end{itemize}
\end{itemize}
\textbf{end}

\textbf{else}

\begin{itemize}
  \item \textbf{if} \( S[i,j] = S[i-1,j] \) \textbf{then}
    \begin{itemize}
      \item \textsc{Diagonalalignmentoutput}(X,Y,S,i-1,j)
      \item print \((x_i)\)
    \end{itemize}
  \item \textbf{else if} \( S[i,j] = S[i,j-1] - \beta \) \textbf{then}
    \begin{itemize}
      \item \textsc{Diagonalalignmentoutput}(X,Y,S,i,j-1)
      \item print \((\phi_y)\)
    \end{itemize}
  \item \textbf{else}
    \begin{itemize}
      \item \textsc{Diagonalalignmentoutput}(X,Y,S,i-1,j-1)
      \item print \((x_i)\)
    \end{itemize}
\end{itemize}
\textbf{end}

\textbf{end}

Figure 4.15: Traceback procedure for delivering an optimal DA.

for the indel such that the flow of movements become closer to the diagonal. Under the homogeneous sequences, it is expected that the same nucleotides are located on the diagonal of the alignment matrix. Therefore, any of them that are closer to the diagonal should have less weight than the other. We define the weight according to the position and direction of each cell. In the upper part, we give lower weight to the horizontal movement because we reach to the diagonal sooner than vertical movement. In the lower part, we give lower weight to the vertical movement, because we reach to the diagonal sooner than horizontal movement.

In the diagonal alignment, we draw two diagonals. The diagonals start from the leftmost cell of the first row and from the rightmost cell of the last row. The table will be divided in three parts: upper part, lower part and between the diagonals. Any of the movement that is closer to diagonal has no penalty. In the upper part, we do not give the penalty to the horizontal movement, between the diagonals there is no priority and in the lower part, we do not give the penalty to the vertical movement.
Chapter 5

Simulation

This section explores the validity of the proposed algorithms for the sequences with different lengths. Simulation investigations are used to study the finite sample properties of the algorithms. In order to provide a meaningful comparison of various algorithms, the proposed alignment algorithms are simultaneously based on the same data.

In order to make a comparative evaluation of the procedures, we seek the certain desirable features such as similarly under the same ancestor and low similarly under the different ancestors. The same criterion given in (2.3),

\[ S = \alpha N_1 - \gamma N_2 - \beta N_3, \]

that measures the similarity between sequence alignment is used to compare the accuracy of the proposed algorithms. The following section includes the study of \( S \) under the proposed homogeneous and non-homogeneous sequences.

5.1 Homogeneous

Consider an ancestor sequence with the length of \( N = 300 \), denoted as \( A \). In order to study the performance of the proposed algorithms, one needs two homogeneous sequences \( X \) and \( Y \) with lengths of \( m \) and \( n \). In order to obtain the homogeneous sequence \( X \) and \( Y \) from \( A \), we choose \((300 - m)\) random numbers from \( \{1, ..., N\} \) without repetition, then order and delete the corresponding from \( A \), therefore \( X \) is a sequence with the length of \( m \) from \( A \), and in reality it is the same as \( A \) and just \( 300 - m \) elements are deleted. The same procedure is applied to \( Y \). We studied the simulation for the different combinations of \((m, n) = \{(270, 270), (250, 270), (230, 270), (200, 270), (180, 270)\} \). The
sequence alignments are applied with different penalty costs, i.e., $\beta = 2, 4, 6, 8$. Tables 5.1-5.5 include the simulations of $S$ under the homogeneity of the sequences.

The simulations given in the Tables 5.1-5.5 show the similarity criterion, $S$, under the proposed algorithms. Obviously, the values of $S$ are quite close to each other. Therefore the GA, WA and DA has the same performance under the homogeneous sequences.

### 5.2 Non-homogeneous

It is important to have an algorithm that is sensitive to the non-homogeneous sequences. In order to obtain the non-homogeneous sequences, a random sample of size $n$ and $m$ are obtained without replacement from $A$, i.e., one selects $n$ and $m$ random numbers from \{1, $\ldots$, $N$\} without replacement and select the corresponding random numbers from the elements of $A$. Obviously these sequences do not have the order of the elements of $A$ and are generated with random elements. Since these sequences are non-homogeneous, one expects the accurate proposed algorithms lead to the lower value score of $S$.

Tables 5.6-5.10 include the results of simulations under the non-homogeneous sequences. Since these sequences are not homogeneous, we expected a lower value for the $S$ under the proposed algorithms. Obviously the lower one provides the accurate algorithm to diagnose the non-homogeneous sequences. The results are quite interesting. In Table 5.6, for $\alpha = \gamma = 8$ and $\beta = 8, 6$, DA works better but for $\beta = 2$, WA leads to the better result. Analyzing the other tables, it makes obvious that the robustest result is obtained under $\beta = 2, 4$ and WA. Thus it works better than the other methods regardless of $\alpha$, $\gamma$, $n$ and $m$. Hence WA can be nominated as the algorithm to achieve the alignment.
Table 5.1: The simulation of $S$ under the homogeneous sequences for $m = 270$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
<th>(8,8)</th>
<th>(8,6)</th>
<th>(8,4)</th>
<th>(8,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>8</td>
<td>1598.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1598.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1549.8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>1678.2</td>
<td>1688.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1678.1</td>
<td>1688.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1657.1</td>
<td>1674.9</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>1758.3</td>
<td>1768.5</td>
<td>1778.7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1758.1</td>
<td>1768.1</td>
<td>1778.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1757.7</td>
<td>1757.7</td>
<td>1769.7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>1858.7</td>
<td>1858.7</td>
<td>1858.7</td>
<td>1868.9</td>
<td></td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1858.0</td>
<td>1858.0</td>
<td>1858.0</td>
<td>1868.5</td>
<td></td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1858.7</td>
<td>1858.7</td>
<td>1858.7</td>
<td>1864.7</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.2: The simulation of $S$ under the homogeneous sequences for $m = 250$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
<th>(8,8)</th>
<th>(8,6)</th>
<th>(8,4)</th>
<th>(8,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>8</td>
<td>1373.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1372.9</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1372.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>1471.1</td>
<td>1485.9</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1470.7</td>
<td>1485.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1462.1</td>
<td>1485.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>1569.1</td>
<td>1584.0</td>
<td>1598.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1568.7</td>
<td>1583.4</td>
<td>1598.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1569.0</td>
<td>1579.5</td>
<td>1598.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>1696.8</td>
<td>1696.8</td>
<td>1696.8</td>
<td>1711.6</td>
<td></td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1695.8</td>
<td>1695.8</td>
<td>1695.8</td>
<td>1711.0</td>
<td></td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1696.8</td>
<td>1696.8</td>
<td>1696.8</td>
<td>1711.5</td>
<td></td>
</tr>
</tbody>
</table>
Table 5.3: The simulation of $S$ under the homogeneous sequences for $m = 230$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
<th>$(8,8)$</th>
<th>$(8,6)$</th>
<th>$(8,4)$</th>
<th>$(8,2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>8</td>
<td>1138.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1138.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1138.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>1260.0</td>
<td>1276.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1259.3</td>
<td>1275.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1254.5</td>
<td>1275.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>1380.5</td>
<td>1397.9</td>
<td>1414.4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1379.8</td>
<td>1397.0</td>
<td>1413.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1380.5</td>
<td>1395.3</td>
<td>1414.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>1534.5</td>
<td>1533.5</td>
<td>1534.4</td>
<td>1552.8</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1531.6</td>
<td>1531.6</td>
<td>1533.3</td>
<td>1551.6</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1534.5</td>
<td>1533.5</td>
<td>1534.4</td>
<td>1552.7</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 5.4: The simulation of $S$ under the homogeneous sequences for $m = 200$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
<th>$(8,8)$</th>
<th>$(8,6)$</th>
<th>$(8,4)$</th>
<th>$(8,2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>8</td>
<td>770.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>769.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>769.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>931.2</td>
<td>947.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>930.4</td>
<td>946.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>927.8</td>
<td>946.8</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>1092.2</td>
<td>1108.9</td>
<td>1128.4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1091.2</td>
<td>1107.2</td>
<td>1127.7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1092.1</td>
<td>1107.1</td>
<td>1128.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>1289.0</td>
<td>1289.8</td>
<td>1290.0</td>
<td>1306.4</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>1285.3</td>
<td>1285.6</td>
<td>1285.1</td>
<td>1303.9</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td></td>
<td>1289.0</td>
<td>1289.8</td>
<td>1290.0</td>
<td>1306.3</td>
<td>-</td>
</tr>
</tbody>
</table>
Table 5.5: The simulation of $S$ under the homogeneous sequences for $m = 180$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
<th>(8,8)</th>
<th>(8,6)</th>
<th>(8,4)</th>
<th>(8,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>8</td>
<td>514.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>514.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>514.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>707.0</td>
<td>723.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>705.5</td>
<td>722.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>703.9</td>
<td>723.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>899.4</td>
<td>915.7</td>
<td>931.7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>897.2</td>
<td>913.4</td>
<td>930.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>899.3</td>
<td>914.2</td>
<td>931.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>1124.5</td>
<td>1124.5</td>
<td>1124.2</td>
<td>1140.6</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>1119.3</td>
<td>1119.3</td>
<td>1118.7</td>
<td>1135.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>1124.5</td>
<td>1124.5</td>
<td>1124.2</td>
<td>1140.5</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 5.6: The simulation of $S$ under the non-homogeneous sequences for $m = 270$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
<th>(8,8)</th>
<th>(8,6)</th>
<th>(8,4)</th>
<th>(8,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>8</td>
<td>176.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>-35.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>-1230.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>379.2</td>
<td>473.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>94.0</td>
<td>226.8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>-175.2</td>
<td>-140.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>605.7</td>
<td>685.1</td>
<td>771.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>336.2</td>
<td>387.9</td>
<td>507.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>411.5</td>
<td>411.6</td>
<td>465.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>994.3</td>
<td>994.3</td>
<td>994.3</td>
<td>1074.4</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>748.3</td>
<td>748.3</td>
<td>748.3</td>
<td>816.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>970.2</td>
<td>970.2</td>
<td>970.2</td>
<td>1010.5</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Table 5.7: The simulation of $S$ under the non-homogeneous sequences for $m = 250$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
<th>(8,8)</th>
<th>(8,6)</th>
<th>(8,4)</th>
<th>(8,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>8</td>
<td>158.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>-38.2</td>
<td>158.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>50.5</td>
<td>158.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>358.3</td>
<td>446.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>94.2</td>
<td>223.0</td>
<td>446.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>176.2</td>
<td>369.3</td>
<td>446.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>580.7</td>
<td>655.4</td>
<td>738.1</td>
<td>1029.8</td>
<td>1029.8</td>
</tr>
<tr>
<td>WA</td>
<td>329.9</td>
<td>372.8</td>
<td>492.2</td>
<td>1029.8</td>
<td>1029.8</td>
<td></td>
</tr>
<tr>
<td>DA</td>
<td>550.9</td>
<td>556.7</td>
<td>690.0</td>
<td>1029.8</td>
<td>1029.8</td>
<td></td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>954.4</td>
<td>952.9</td>
<td>954.7</td>
<td>1009.6</td>
<td>1009.6</td>
</tr>
<tr>
<td>WA</td>
<td>723.3</td>
<td>728.9</td>
<td>720.4</td>
<td>786.6</td>
<td>786.6</td>
<td></td>
</tr>
<tr>
<td>DA</td>
<td>940.3</td>
<td>938.6</td>
<td>939.3</td>
<td>1009.6</td>
<td>1009.6</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.8: The simulation of $S$ under the non-homogeneous sequences for $m = 230$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
<th>(8,8)</th>
<th>(8,6)</th>
<th>(8,4)</th>
<th>(8,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>8</td>
<td>124.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>-45.5</td>
<td>124.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>4.04</td>
<td>124.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>324.6</td>
<td>407.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>94.2</td>
<td>211.4</td>
<td>407.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>141.6</td>
<td>321.3</td>
<td>407.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>544.0</td>
<td>614.9</td>
<td>691.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WA</td>
<td>322.6</td>
<td>363.7</td>
<td>478.1</td>
<td>691.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>494.5</td>
<td>512.6</td>
<td>639.2</td>
<td>691.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>908.0</td>
<td>908.0</td>
<td>907.9</td>
<td>979.6</td>
<td>979.6</td>
</tr>
<tr>
<td>WA</td>
<td>706.2</td>
<td>706.2</td>
<td>706.2</td>
<td>767.2</td>
<td>767.2</td>
<td></td>
</tr>
<tr>
<td>DA</td>
<td>883.9</td>
<td>883.9</td>
<td>883.9</td>
<td>958.3</td>
<td>958.3</td>
<td></td>
</tr>
</tbody>
</table>
Table 5.9: The simulation of $S$ under the non-homogeneous sequences for $m = 200$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(8,8)</td>
</tr>
<tr>
<td>GA</td>
<td>8</td>
<td>34.5</td>
</tr>
<tr>
<td>WA</td>
<td>-96.4</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>-57.6</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>246.7</td>
</tr>
<tr>
<td>WA</td>
<td>61.6</td>
<td>167.4</td>
</tr>
<tr>
<td>DA</td>
<td>99.3</td>
<td>252.7</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>473.6</td>
</tr>
<tr>
<td>WA</td>
<td>304.9</td>
<td>336.5</td>
</tr>
<tr>
<td>DA</td>
<td>416.5</td>
<td>450.7</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>825.2</td>
</tr>
<tr>
<td>WA</td>
<td>670.5</td>
<td>670.5</td>
</tr>
<tr>
<td>DA</td>
<td>802.2</td>
<td>802.2</td>
</tr>
</tbody>
</table>

Table 5.10: The simulation of $S$ under the non-homogeneous sequences for $m = 180$ and $n = 270$

<table>
<thead>
<tr>
<th>method</th>
<th>$\beta$</th>
<th>$(\alpha, \gamma)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(8,8)</td>
</tr>
<tr>
<td>GA</td>
<td>8</td>
<td>-55.8</td>
</tr>
<tr>
<td>WA</td>
<td>-165.2</td>
<td>-</td>
</tr>
<tr>
<td>DA</td>
<td>-114.0</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>6</td>
<td>172.0</td>
</tr>
<tr>
<td>WA</td>
<td>23.6</td>
<td>111.5</td>
</tr>
<tr>
<td>DA</td>
<td>62.1</td>
<td>193.2</td>
</tr>
<tr>
<td>GA</td>
<td>4</td>
<td>411.0</td>
</tr>
<tr>
<td>WA</td>
<td>277.2</td>
<td>301.2</td>
</tr>
<tr>
<td>DA</td>
<td>360.0</td>
<td>401.8</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>758.2</td>
</tr>
<tr>
<td>WA</td>
<td>633.9</td>
<td>633.9</td>
</tr>
<tr>
<td>DA</td>
<td>741.1</td>
<td>741.1</td>
</tr>
</tbody>
</table>
Chapter 6

Conclusion and Future Work

This work explores two new alignment algorithms. The new proposed, weighted alignment and diagonal alignment provide the same result as the existing algorithms under the homogeneous sequences, but under the non-homogeneous sequences, the proposed algorithms provide the score with less values. The two proposed algorithms are more accurate than the existing approaches. We have applied our new algorithms on several examples and the results are promising.

The simulation clarifies that the choice of $\beta$ affects the alignment and the small value of it under dissimilarity leads to the small value of $S$, hence finding the optimal value for $\beta$ is too much important, it is under study by the author.

This thesis explored the paired comparisons, and the multiple alignments is of interest to be studied. Another interests are to study the new algorithms for the local alignment and the Genome alignment.
Bibliography


