Problem Chosen	MCM/ICM Summary Sheet	Team Control Number
B		MS23293

Use this template to begin typing the first page (summary page) of your electronic report. This template uses a 12-point Times New Roman font. Submit your paper as an Adobe PDF electronic file (e.g. 1111111.pdf), typed in English, with a readable font of at least 12-point type.

Do not include the name of your school, advisor, or team members on this or any page.

Papers must be within the 25 page limit.

Be sure to change the control number and problem choice above.
You may delete these instructions as you begin to type your report here.

Follow us @COMAPMath on Twitter or COMAPCHINAOFFICIAL on Weibo for the most up to date contest information.

Contents

1 Introduction

1.1 Problem Background

Ribonucleic acid (RNA) is a polymeric molecule essential in various biological roles in coding, decoding, regulation and expression of genes. RNA and deoxyribonucleic acid (DNA) are nucleic acids. Along with lipids, proteins, and carbohydrates, nucleic acids constitute one of the four major macromolecules essential for all known forms of life. Like DNA, RNA is assembled as a chain of nucleotides, but unlike DNA, RNA is found in nature as a single strand folded onto itself, rather than a paired double strand. [1]

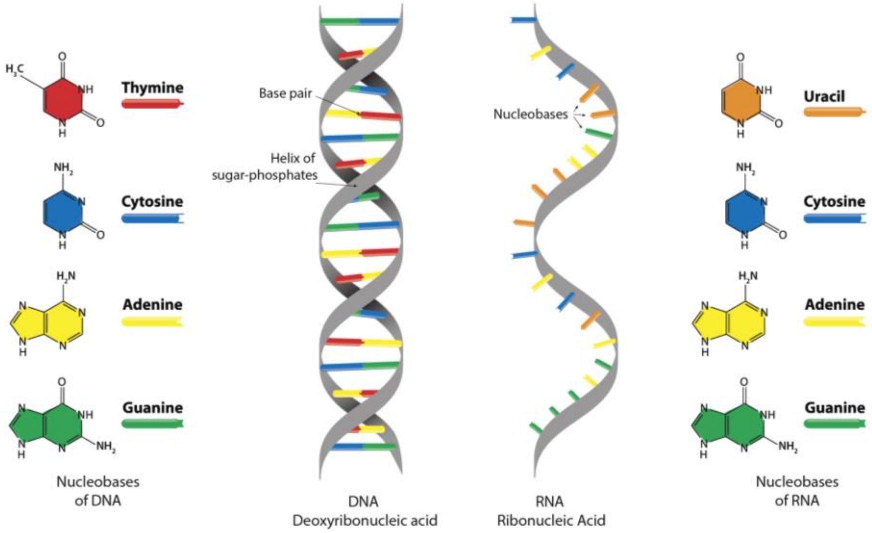


Figure 1: The differences of structure between DNA and RNA [2]

Homology is an important concept in bioinformatics research. It refers to the relationship between branches that evolved from a common ancestor, the essential similarity arising from a common evolutionary or ontogenetic source. At the microscopic level, homology is usually expressed as similarity between nucleotide sequences of two nucleic acid molecules. In the genetic process of RNA sequence, there will be some differences between different sequences due to the mutation of the base. These mutations can be classified into three types: replacement, insertion, and deletion. In substitution, a purine is replaced by another purine or a pyrimidine by another pyrimidine. Insertion may be caused by the addition of one or more bases. Deletion usually takes the form of deleting one or more bases. We can measure the distance between two sequences by the number of mutations. Multiple sequences of bases close together can form a family, and they can be considered homologous. Once homologous sequences are found, a more accurate alignment can be established through multiple sequence alignments, laying the foundation for subsequent phenotypic prediction and evolutionary analysis.

1.2 Restatement of the Problem

Considering the background information and restricted conditions given in the problem statement, we are supposed to solve the problems below:

- Design a model which can quickly compute two basic sequences containing at least 10^3 bases.

- Evaluate the accuracy and complexity of the model established in Question 1 and use appropriate examples to demonstrate it.
- If multiple base sequences ina family evolved from the same ancestral sequence, design a model to determine their ancestral sequence and draw a lineage tree.

1.3 Our Approach

In our model, we first establish a Dynamic Programming Model and use Levenshtein Distance Algorithm to measure the distance between two sufficient long base sequences. Next, we evaluate the complexity of the algorithm and optimize it. Then, we evaluate the accuracy of the model I in different situations. Besides, we take RNA sequences of COVID-19 from professional database for instance and to illustrate
After that, we

2 General Assumptions

Our models rely on the following assumptions. Some assumptions are throughout the text. These assumptions will simplify the problem. Other assumptions may not be as follows, but will be put forward in the model push.

- **It is assumed that mutation is caused by the change of a single base.** In reality, there will also be a frameshift mutation, which refers to a mutation that causes the dislocation of a series of coding sequences after the insertion or loss of one or more base pairs at a certain site in a DNA fragment. It can cause abnormalities in genetic information beyond the site. The mutation is so damaging to genetic information that it can even lead to the death of the next generation. Therefore, we will not discuss these changes in this article.
- **It is assumed that both base sequences to be tested are valid sequences.** Each base in its sequence can be distinguished, and only the continuous arrangement of" ATCG" can appear in the coding, without N value.

3 Notations

Table 1: Notations

Symbol	Description
$LevD_{X,Y}$	Levenshtein distance between sequence X and Y
$GLD_{X,Y}$	Normalization distance between sequence X and Y
$D_{MEGA}(X,Y)$	Distance that the software MEGA obtains between sequence X and Y
c	Mutation rate
A	Accuracy
$\|X\|$	The length of sequence X

4 Model I: Sequence Distance Measurement Model

4.1 Model Preparation

Through the detection of modern scientific and technological means, the genetic information of RNA is usually converted into a series of base sequence coding for representation and recording. Since RNA is not easy to preserve and extract, scientists often reverse transcript it into DNA, and routinely use DNA instead when talking about genetic coding. As a result, for any code representing RNA's genetic information, the only possible letters are A, C, G, and T, which stand for the four nucleotides that make up DNA -- adenine, cytosine, Guanine, thymine. Each letter represents a base, and they are arranged in unspaced rows to form a string such as "AAAGTCTGAC".

Taking the above information into consideration, the three kinds of mutations of RNA base sequences can be described as three kinds of editing for string: insertion, replacement or deletion respectively. As a result, we can use Levenshtein distance to measure the distance between two base sequences.

Levenshtein distance is a frequent-used type of edit distance. Levenshtein distance between two strings is the minimum number of single-character edits (insertions, deletions and substitutions) required to change one word into the other. It is frequently used in bioinformatics to analyze the similarity of two base sequences.

4.2 Model building

Assuming that the string encoded by the source base sequence has n characters and the string encoded by the target base sequence has m characters, we need to find the minimum number of edits required to convert n characters from the source string to m characters from the target string.

Let's assume that X, Y are two base sequences. $X = \{x_1, x_2 \ldots x_n\}$and $Y = \{y_1, y_2 \ldots y_m\}$. $(n, m \geq 1000)$. The Levenshtein distance between X, Y can be expressed as LevD$_{X,Y}$. More generally, the Levenshtein distance between the first i letter of X and the first j letter of Y can be expressed as $LevD_{X,Y}[i][j]$ $(i \leq n, j \leq m)$.

It is apparent that if $i = 0$, $LevD_{X,Y}[0][j] = j$, which means that we need to insert j letters in X or delete j letters in Y to make X and Y the same. Similarly, if $j = 0$, $LevD_{X,Y}[i][0] = i.$

In order to calculate $LevD_{X,Y}[i][j](0 < i \leq n, 0 < j \leq m)$, we need to know $LevD_{X,Y}[i-1][j]$, $LevD_{X,Y}[i][j-1]$, and $LevD_{X,Y}[i-1][j-1]$.

- If the $i'th$ letter of X is the same as the $j'th$ letter of Y, $LevD_{X,Y}[i][j] = LevD_{X,Y}[i-1][j-1]$.
- If the $i'th$ letter of X is different from the $j'th$ letter of Y, there will be three ways to make them the same.
    a) Insertion: Insert the $j'th$ letter of Y at the end of $\{x_1, x_2, \ldots x_{i-1}\}$, then the $i'th$ letter of X is the same as the $j'th$ letter of Y, $LevD_{X,Y}[i][j] = 1 + LevD_{X,Y}[i-1][j]$.
    b) Deletion: Delete the $i'th$ letter of X in $\{x_1 x_2, \ldots x_i\}$, then $LevD_{X,Y}[i][j] = 1 + LevD_{X,Y}[i][j-1]$.
    c) Replacement: Replace the $i'th$ letter of X with the $j'th$ letter of Y, then $LevD_{X,Y}[i][j] = 1 + LevD_{X,Y}[i-1][j-1].$

  As a result, if the $i'th$ letter of X is different from the $j'th$ letter of Y, $LevD_{X,Y}[i][j] = 1 + min\{LevD_{X,Y}[i-1][j-1], LevD_{X,Y}[i-1][j], LevD_{X,Y}[i][j-1]\}$.

The above formula is not difficult to find by using recursive functions, but direct recursion will cause a lot of double computation. So, it is better to use loops and matrices combined with dynamic programming model to solve the problem.

We use a $(n+1)\times(m+1)$ matrix $LD[0\dots n, 0\dots m]$ to save the$LevD_{X,Y}[i][j]$ that has been calculated at present, so that the data can be directly read from the matrix when it is needed for later calculation.

Algorithm 1.1: Distance Measurement

```
Input: $X, Y$
Output: $LevD_{X,Y}[n][m]$
for $j = 0\ to\ m$ do
    for $i = 0\ to\ n$ do
        Compare $x_i$ with $y_j$
        Calculate Lev$D_{X,Y}[i][j]$
        Record the results in $n \times m$ matrix LD at position$(i, j)$
    end
end
```

4.3 Description of the Distance

Assume that there are two base sequences X, Y, and the length of base sequences can be expressed as$|X|, |Y|$. According to the algorithm in 4.2, the Levenshtein distance between X and Y can be expressed as $LevD_{X,Y}$. However, since the length of sequence isn't considered about, $LevD_{X,Y}$ cannot reflect the similarity of two sequences properly.

Thus, we can use normalization distance to measure the similarity. The usual way to construct normalization distance is

$$NLD_{X,Y} = \frac{2LevD_{X,Y}}{|X|+|Y|} \qquad (1)$$

However, this kind of normalization distance does not satisfy the properties of the triangular inequality, which is also the most important property of Levenshtein distance: Assuming that there are three sequences $X, Y, and\ Z$, $LevD_{X,Y} + LevD_{Y,Z} \geq LevD_{X,Z}$. As a result, NLD is not proper to build genealogical tree.

A better way to construct the normalization distance is

$$GLD_{X,Y} = \frac{2LevD_{X,Y}}{\alpha \cdot (|X|+|Y|) + LevD_{X,Y}} \qquad (2)$$

α is a coefficient, $|X|\ and\ |Y|$ is the length of X and Y, $LevD_{X,Y}$ is the Levenshtein distance between X and Y. The advantage of this construction is that $GLD_{X,Y}$ follows the properties of the triangular inequality, so the distance can be used in building genealogical tree. In our article, we use GLD to measure the distance of two base sequences.

4.4 The Determination of the Value of α

4.4.1 Definition of Accuracy

We use the professional gene sequence distance measurement software MEGA to calculate the distance and believe that the distance D_{MEGA} obtained is the real distance between the sequences. We believe that accuracy can be evaluated by the component of the difference between the predicted and ideal values. In the article, the accuracy will be expressed as

$$A = 1 - \left|\frac{D_{MEGA}(X,\ Y) - \mathrm{GLD}_{X,\ Y}}{D_{MEGA}(X, Y)}\right| \qquad (3)$$

The result obtained here is in the interval [0,1]. The larger the result obtained here is, the higher the accuracy is.

4.4.2 Looking for the most appropriate value of α

We randomly generate a base sequence with the length of 1200, and made some editing (including insertion, deletion and replacement), and then calculate the exact distance D_{MEGA} between the two sequences with the software MEGA. At the same time, we also use the algorithm in 4.2 to obtain the Levenshtein distance $LevD_{X,Y}$, and substitute it into Equation (2) to obtain the normalized distance $GLD_{X,Y}$. The accuracy of $GLD_{X,Y}$ compared to D_{MEGA} is calculated by Equation (3) when the value of α is varied in the range of 1.30 to 1.70.
Do the same thing when the sequence length is 2000 and 3000 and draw Figure 2.

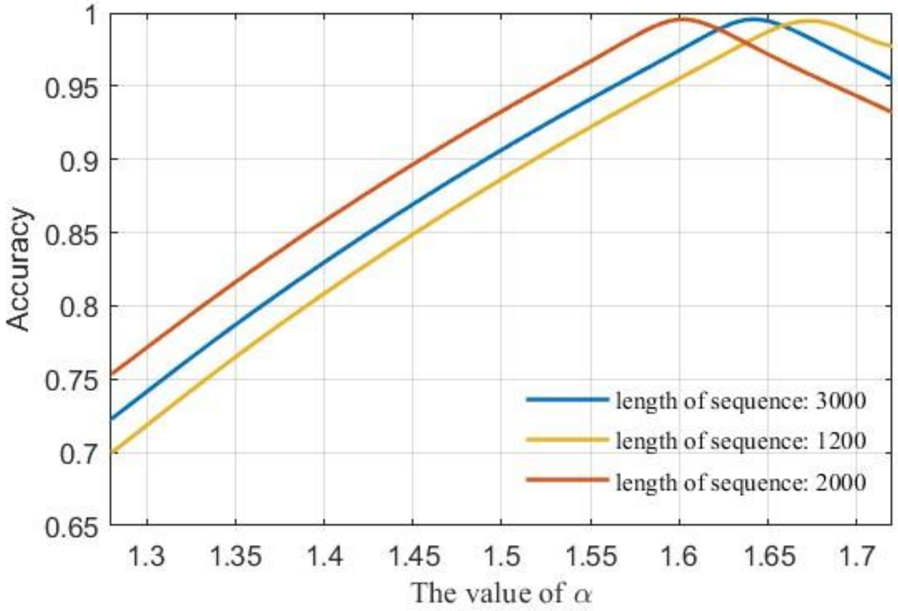


Figure 2: Trend of the accuracy with the value of α

From Figure 2, we can learn that even though the sequence length changes, the accuracy can be kept above 0.9 when the value of α is range from1.60 to1.68. Thus, we assume that $\alpha = 1.64$ is the most appropriate. Equation (2) can be re-written as

$$GLD_{X,Y} = \frac{2LevD_{X,Y}}{1.64 \cdot (|X| + |Y|) + LevD_{X,Y}} \qquad (4)$$

4.5 Model Evaluation and Majorization of Complexity

4.5.1 Evaluation of Complexity on Algorithm 1.1

The algorithmic complexity is mainly about the amount of computer resources needed to run the algorithm. We usually use $T(n) = O\big(f(n)\big)$ to represent the time complexity, and $S(n) = O\big(f(n)\big)$ to represent the space complexity.
In Algorithm 1.1, the $n + 1 times$ cycle is nested in the $m + 1 times$ cycle, so the time complexity can be expressed as $T(n) = O\big((m+1) * (n+1)\big) = O(m * n + m + n + 1)$. Since

$m, n \geq 1000$, $m * n$ is far larger than $m + n + 1$, and Time complexity can be simplified as $O(m * n)$.
In Algorithm 1.1, there is a $(n + 1) \times (m + 1)$ matrix, so the space complexity can be expressed as $S(n) = O\big((m + 1) * (n + 1)\big)$, which can also be simplified as $O(m * n)$.
In conclusion, both the time complexity and the space complexity of Algorithm will increase more and more rapidly as m and n increase.

4.5.2 Majorization

As m and n increase, the matrix becomes more and more redundant. Thus, we can transform the $(n + 1) \times (m + 1)$ matrix into the $(n + 1) \times 2$ matrix.

Algorithm 1.2: Distance Measurement

```
Input: $X, Y$
Output: $LevD_{X,Y}[n][m]$
for $j = 0\ to\ m$ do
  | Mark the number at the position (0,2) of the matrix as $j + 1$
  for $i = 0\ to\ n$ do
  | Compare $x_i$ with $y_j$
  | Calculate $LevD_{X,Y}[i][j]$
  | Record the results in $(n + 1) \times 2$ matrix LD at position$(i + 1,2)$
  end
  for $i = 0\ to\ n$ do
  | Mark the number at the position $(i + 1,1)$ of the matrix as the number at position$(i + 1,2)$
  end
end
```

4.5.3 Evaluation of Complexity on Algorithm 1.2

In Algorithm 1.2, the time complexity is the same as that of Algorithm 1.1. It can be expressed as $O(m * n)$. The space complexity decreases from $O(m * n)$ to $O(n)$, which means that it changes from square growth to linear growth as m, n increases.

5 Model II: Model Evaluation on Accuracy

5.1 Model Preparation

5.1.1 Overview

This model is used to evaluate the final optimization model obtained in Model 1. Similar to the procedure for finding the best alpha value, we take any sufficiently long sequence X, Y(the length of X, Y respectively is greater than 1000). We use MEGA to calculate the distance $D_{MEGA}(X,\ Y)$ between X and Y as the ideal value. $GLD_{X,Y}$ is calculated by using Model 1. Then, we evaluate the accuracy in different situations.

5.1.2 Definition of Mutation Rate

We defined the mutation rate c as the percentage of the number of modifications in the total length of the original sequence:

$$c = \frac{\text{the number of modifications}}{\text{the total length of the original sequence}} \times 100\%$$

5.2 Evaluation Process

5.2.1 Mutation rate is fixed and change the sequence length.

A sequence X_{2000} with a length of 2000 bases was randomly generated, and the mutation rate was set as 5% (i.e., 20 replacements, 40 deletions, and 40 additions) to obtain a new sequence Y_{2000}. Calculate the accuracy A. Experiment n times, and get:

$$A_{2000} = \frac{1}{n}\sum_{i=1}^{n} A_i$$

Do the same thing for A_{1200}, A_{1400}……A_{3000} and draw Figure 3.

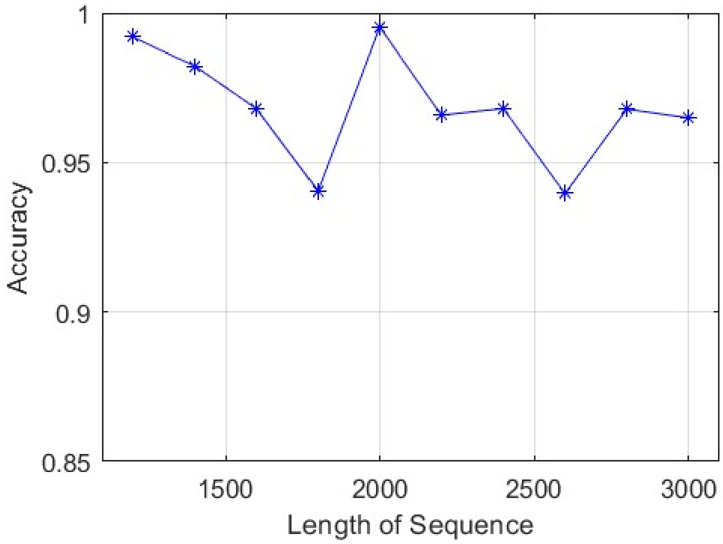


Figure 3: Trend of Accuracy with length of sequence

According to the above analysis, when the variation rate is constant, the accuracy fluctuates with length, but the accuracy is stable above 94%, and the average accuracy is 96.84%.

5.2.2 Sequence length is fixed and change the mutation rate.

The sequence $X_{5\%}$ with A length of 2000 bases is randomly generated, and the variation rate was set as 5% (i.e., 40 modifications, 30 deletions and 30 additions) to obtain a new sequence $Y_{5\%}$. Calculate the accuracy A. Experiment n times, and get:

$$A_{5\%} = \frac{1}{n}\sum_{i=1}^{n} A_i$$

Do the same thing for $A_{2.5\%}$, $A_{7.5\%}$……$A_{17.5\%}$ and draw Figure 4.

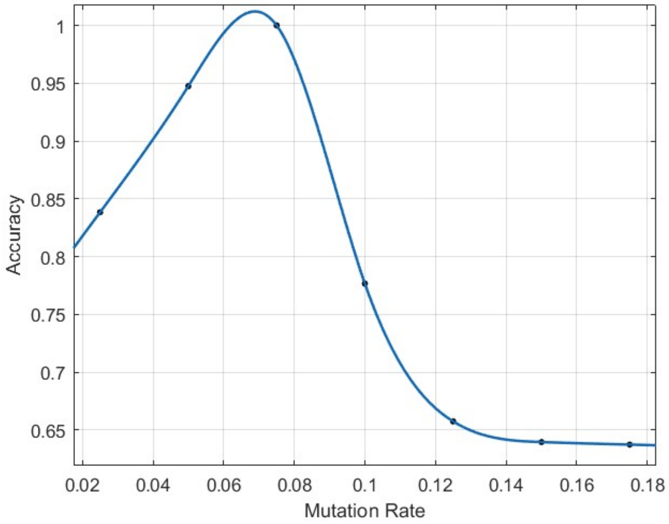


Figure 4: Trend of Accuracy with Mutation Rate

By analyzing Figure 4, it can be concluded that when the sequence length remains unchanged and the variation rate changes, the accuracy will change significantly, with the specific trend of increasing first and then decreasing, reaching the peak when the variation rate is about 7.5%. Then it will decline rapidly and level off when the variation rate is greater than 12.5%. Accuracy will not fall below 60%. The data in Figure 4 is consistent with that in 5.2.1, indicating high reliability.

5.3 Examples: Take COVID-19 for Instance

The model is used to analyze the distance between two sequences of COVID-19. The data comes from COVID-19 Data Portal [4]. One sequence is the Delta strain, while the other is the Alpha strain. The length of each sequence is around 30000, and the mutation rate is around 4%.
It is found that $\mathrm{GLD}_{X,Y}$ is 0.01556 and $D_{MEGA}(X,\ Y)$ is 0.018, so the accuracy is 86.4%. This result accords with the conclusion of the above two accuracy analyses.

5.4 Conclusion

From the above accuracy analysis, it can be concluded that the accuracy of this model is less affected by the sequence length and more affected by the variation rate. This indicates that the model is suitable for sequences of biologically close groups of organisms. The model has high accuracy in the determination of homologous sequence and homologous sequence distance.

6 Model III: Building Genealogical Tree

6.1 Building Genealogical Tree Model based on UPGMA

UPGMA is an arithmetic averaging unweighted group pair algorithm. It is used in numerical taxonomy to reflect the degree of similarity in the representation of groups. We assume that the rate of gene mutation is constant. With this algorithm, we can construct molecular system trees. To facilitate analysis, we need to define a continuous sum-distance function first. We define that the distance between two taxa is proportional to the total length of the branches connecting the two taxa in the phylogenetic tree. In this case, the distance between taxon a and taxon b is $d_{av}+d_{bv}$ if the taxon A and taxon B are connected by two sides of length d_{av} and d_{bv} which passing through an intermediate node v.

The basic idea of UPGMA is to first select the pair of taxa (RNA sequence) with the smallest distance from the distance matrix, naming them x and y respectively, and then combine the two taxa to form a new taxon z, which are regarded as their ancestors. Recalculate the distance $d(z,u)$ from the new taxon u. Its calculation formula is:

$$d(z,u) = \frac{1}{2}\big(d(x,u) + d(y,u)\big) \tag{5}$$

The new taxon formed by each merge is actually a cluster, which is represented by an internal node. This node is the same distance from the node where x and y are located, and its value is equal to half of $d(x,y)$. The distance to other nodes is calculated according to the above formula. After each merge, we need to modify the distance matrix. The process is repeated until all taxa have been combined into one category.

The UPGMA process is as follows:

1. Initialization: make each taxonomic unit a class of its own. If there are n taxonomic units, there are n classes at the beginning. The size of each class is 1, and each class is represented by n leaf nodes.
2. Execute the following cycle:

Step 1: Find two classes x and y with minimum distance in the distance matrix.

Step 2: A new cluster u was established as the merge of taxon x and taxon y.

Step 3: Create a new internal node u in the tree, generate two new branches. Connecting x and y to u, making u be the parent node of x and y, and connect u to each leaf of x and y. The length of the node is assigned as $\frac{1}{2}GLD_{x,y}$ (The calculation and definition of GLD have been given in Model 1), so $d_{u,x}$ and $d_{u,y}$ are calculated.

Sep 4: Calculate the distance between u and other taxon k according to formula:

$$GLD_{u,k} = \frac{1}{rs}\sum_{i,j} GLD_{i,j} \tag{6}$$

Here, r and s respectively represent the number of classes in taxon u and k, and ij d is the distance between class i in taxon u and class j in taxon k.

Step 5: Update the distance matrix. Delete the row and column corresponding to x and y in the distance matrix, add new rows and columns for u, and establish a new distance matrix.

Repeat the loop until there is only one class left.

Then, the result is a clustering for all of the sequences, from which we can draw a rooted tree. The rooted tree is the genealogical tree that we are asked to draw.

6.2 Ancestry Sequence Searching Model Based on LCS

6.2.1 Definition of Longest Common Subsequence

Assume that X, Z are two sequences, and $X = \{x_1, x_2, \dots x_m\}$, $Z = \{z_1, z_2, \dots z_k\}$.
If there is a strictly increasing subscript sequence $\{i_1, i_2, \dots i_k\}$ which can make each $j = 1, 2, \dots k$ satisfying $z_j = x_{i_j}$, Z is known as the subsequence of X

If Z is the subsequence of X and the subsequence of Y, Z is known as the common subsequence of X, Y.

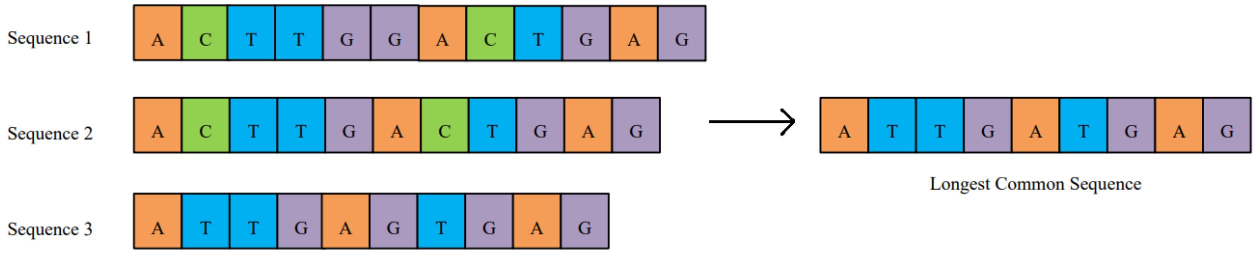
If Z is the longest sequence among all the common subsequence of X and Y, Z is known as the longest common subsequence of X, Y.

6.2.2 Ancestry Sequence Searching

Assume that there are n sequences in the genealogical tree, and they can be expressed as

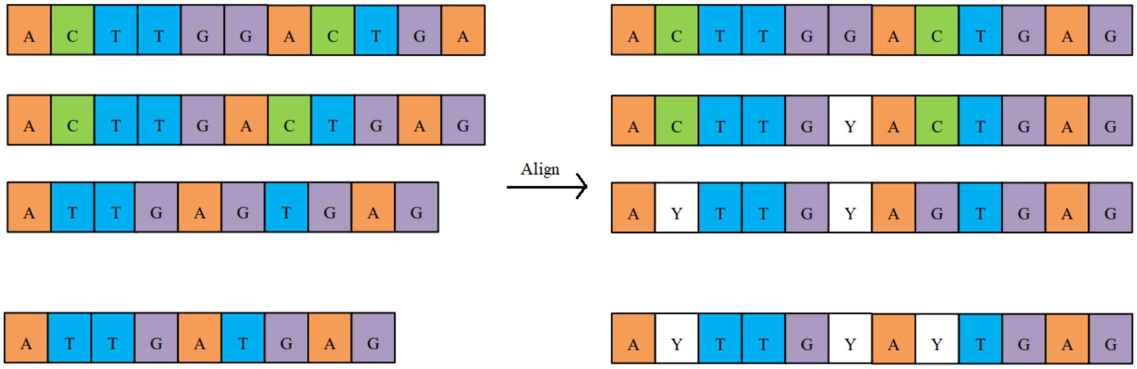
$$S_i, (i = 1,2,3, \dots n)$$

Step 1: Find the longest common subsequence R of $S_i, (i = 1,2,3, \dots n)$

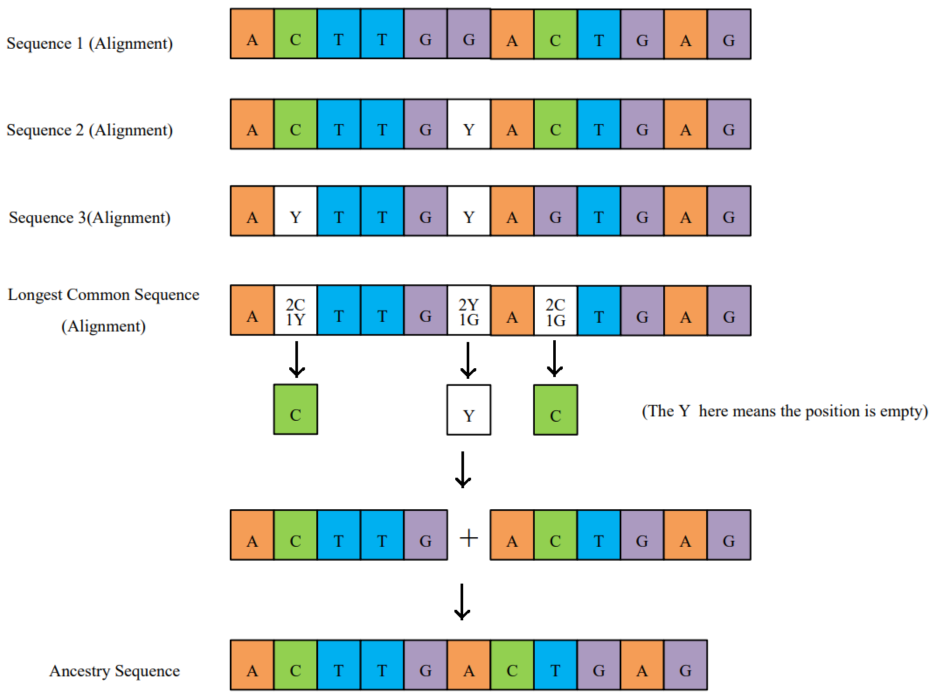


Step 2: Alignment.

The purpose is to make all of the sequences in the same length.
For the first base in R, if the first base in S_1 is not the same as R, then judge the latter base until they are the same at the x_1 base. Do the same thing from $S_2\ to\ S_n$, and get a set$(x_1, x_2, \dots x_n)$.
Add $(max\{x_1, x_2, \dots x_n\} - x_i)$ placeholders 'Y' before the x_ith base of S_i.
Repeat this step until the last base, and now all of the sequences in the same length.



Step 3: For each placeholder Y (assuming the ordinal number is k), make a vertical comparison, that is, take the ordinal number k of all the sequences, and take the most probable occurrence as the base at this position (if the probability of Y is the highest, then take Y).



From the above three steps, we can get an ancestry sequence predicted based on multiple sequences in the genealogical tree.

7 Strengths and Weaknesses

7.1 Strengths

- The space complexity of Algorithm 1.2 in Model I is linear, which means that the algorithm can process longer sequences when the available internal memory is fixed.
- The accuracy of Model I is high and the evaluation of the model on Accuracy demonstrates the effectiveness of the model under different situations when parameters change.
- Trees obtained by the UPGMA method in Model III are usually rooted trees, as roots are easily obtained under the assumption of a constant rate of evolution. Trees constructed by other phylogenetic inference methods are usually rootless trees because it is difficult to identify the roots at different evolutionary rates.

7.2 Weaknesses

- The time complexity of Algorithm 1.1 and Algorithm 1.2 in Model I will both growth at a quadratic square rate when m and n increase.
- The UPGMA method in Model III to build genealogical tree often results in incorrect topological structures when the evolutionary rate of different lineages is very different or parallel evolution with homologous sequences is present. Moreover, when the state space of the evolutionary tree is large, the operability of the UPGMA method is extremely poor, so the use of this tree construction method is extremely limited.

References

[1] https://en.wikipedia.org/wiki/RNA

[2] https://www.technologynetworks.com/genomics/lists/what-are-the-key-differences-between-dna-and-rna-296719
[3] https://zh.wikipedia.org/zh-tw/%E6%A0%B8%E9%85%B8%E5%BA%8F%E5%88%97

[4]COVID-19 Data Portal - accelerating scientific research through data (covid19dataportal.org)

Appendices

Appendix A: Tools and Software

Paper written and generated via Microsoft Word.
Graph generated using MATLAB R2022a.
Codes written in Python3.9.

Appendix B: The Codes