Analysis of Genetic Sequence Homology Based on Dynamic Programming Algorithm

Abstract:

In the evolutionary process of living things, genetic sequences may have evolved from a common ancestral sequence. By analyzing the similarity of genetic sequences, it can be inferred whether two sequences are homologous.

In our paper, we successfully establish an analysis model based on the edit distance algorithm to quantitatively analyze the homology between two genetic sequences. The basic idea of this model is to adopt dynamic programming to calculate distance between sequences.

In the first part, we approximate the mutations and distance between the sequences in order to compare the similarity of these sequences. This similarity correlates to transitions or inversions, insertions and deletions of single base that occur in mutations. Therefore, we constructed an analysis model based on the Needleman-Wunsch edit distance algorithm, simulating the mutation process as a string change, and calculating the similarity of two sequences.

In the second part, we use the base sequences of different species and species traits to clarify the reliability of the algorithm. Meanwhile, sequence segmentation method is adopted to calculate average Needleman-Wunsch similarity, as well as to verify model validation.

In the third part, we propose an extension model to infer the ancestral sequence of multiple individuals in a family. Furthermore, this extension model is supported by reliable data in the gene pool. We use the extension model to draw genetic trees of sequences in a family.

Finally, we further discuss how to contrive a better strategy with several improved methods. In this case, we strive to reduce the accidental error and improve accuracy of models used in a large scale.

Key words: sequence homology, dynamic programming, Needleman-Wunsch edit distance algorithm

Contents

I. Introduction

1.1 Background

There exists no doubt in the significance of genetic sequence alignment to bioinformatics analysis. DNA, RNA or protein sequences contain a large amount of biological genetic information, which is now used to study the relationship between species, and is widely applied to technologies such as biology, pharmaceutical and agriculture. The main function of genetic sequence alignment is to determine whether the sequences are homologous. Briefly speaking, sequence homology refers to the biological homology between genetic sequences, that is, whether two genetic sequences have evolved from a common ancestral sequence. On the basis of sequence homology analysis, species classification and evolutionary trees can be constructed.

Therefore, a realistic model for quantitative analysis of sequence homology is required to solve the following questions:
1) How can we propose an approach to describe the similarity between two sequences?
2) How to evaluate the validation of this approach?
3) How can we apply the model to infer the ancestral sequences?

1.2 Restatement of the problem

The problem requires us to develop a model to measure the similarity between two sequences, then use the model to find the ancestral sequence of several base sequences. For a family of multiple base sequence, homologous fragments are altered due to genetic mutations, which occurs with a relatively low probability. Therefore, if the distance between two sequences measured by mutation points is pretty close, we have reason to believe that they have a high degree of similarity. This model needs to take into account changes in base sequences brought about by mutations, so as to accurately find homologous sequences.

II. Assumptions and Justifications

We approximate the whole process as follows:

- Long genetic sequences, such as DNA fragments, can be simulated using strings. Because of the limited number of bases or amino acids, he type of a string element is fixed. For example, the string corresponding to DNA consists of only four characters: ACGT.
- Since mutations may occur during the genetic process, there exists deletion, adding, substitution of characters. Depending on these differences, we can somehow quantify the similarity between the two strings.

- Considering the low probability of mutations, the more similar two strings are, the more likely it is to assume they are homologous. This view can be supported by comparing actual data of close and distant relatives.
- According to the similarity between multiple strings, we can formulate the whole model through character divisions. Each string is divided according to a certain precision to get the substrings, and the similarity between the substrings is compared. The most similar substrings of each part are collated to get the corresponding string of their common ancestor, and the genetic tree is made according to this process.

III. Models

3.1 Basic Model

To solve the three core problems mentioned above, we apply an algorithm based on dynamic programming to solve the edit distance, then calculate similarity between two sequences.

3.1.1 Terms, Definitions and Symbols

Definition 1: Edit Distance

For a given string 'str', define the following three situations as an operation on string 'str':

- Add a single character at a certain position.
- Reduce a single character at a certain position.
- Replace a single character at a certain position.
- For the given two strings 'str1' and 'str2', the edit distance is defined as the minimum number of operations required to change the longer string of 'str1' and 'str2' into another string. For example, for the given two strings str1 = 'abcde', str2 = 'abced', the edit distance between these two strings is 3.

Definition 2: Null Character

For any character in string 'str', if it is a letter, the character is non-null. If it is '-', the character is a null character.

Definition 3: Scoring Strategy

For two characters 'c1', 'c2', we need to ensure that at least one character is not a null character.

- If ‘c1’ and ‘c2’ are consistent, it is a perfect match, then +1 point.
- If ‘c1’ and ‘c2’ are inconsistent, it is a mismatch, then -1 point.
- If one of ‘c1’ and ‘c2’ is a null character, then -1 point.
- We use $\theta(c1, c2)$ to represent the scoring between characters ‘c1’ and ‘c2’.

3.1.2 Assumptions

For two given genetic sequences, assuming that they can be converted to each other through a finite number of insertions, deletions and substitutions of a single base, it can be represented by a string, and its edit distance can be calculated.

3.1.3 The Foundation of Model

Our model is based on the Needleman-Wunsch algorithm.

Wunsch algorithm is a classic global alignment algorithm, which was first proposed b y Saul B. Needleman and Christian D. Wunsch in 1970 to solve the problem of align ment of protein sequences, and is still in use today.
Step 1: Initialize the score matrix.
Step 2: Calculate the score of each locus from top left to bottom right. The score of ea ch locus is related to the score of the top, left and top left positions of its position. As shown in the formula:

$$C(i,j) = max\begin{cases} C(i-1, j-1) + \theta(\mathrm{Q(i)}, \mathrm{P(j)}) \\ C(i-1, j) + \theta(\mathrm{Q(i)}, -) \\ C(i-1, j-1) + \theta(-, \mathrm{P(j)}) \end{cases}$$

Step 3: After the matrix is generated, start traceback. What we need to know is what is the path to the bottom right corner, the path that we want to match

We use the Needleman-Wunsch algorithm to process tow given strings, and get the number(n) of matching characters between the two strings. Let N be the maximum length of the two strings, and we use $q = n/N$ to indicate the similarity degree of the two strings. It can be considered that q can be used to quantitatively measure the degree of similarity between two sequences, which is called Needleman-Wunsch similarity. Among them, q satisfies $q \in [0,1]$, and the more similar the sequences are, the larger q will be.

3.1.4 Solution and Result

1) Based on Python, we use the Needleman-Wunsch algorithm based on dynamic programming to write a program to calculate the similarity of Needleman-Wunsch.

2) Through genetic databases (such as NCBI), we find several genetic sequences (length longer than 103 bases) corresponding to some common dominant traits of different species. Then, we use the xlwings library of Python to read the data, and select one of the sequences to compare with genetic sequences of other species. According to the above analysis, we can calculate the Needleman-Wunsch similarity, write it into the table and draw diagrams.

3) We change the common dominant trait selected in 2), then select other genetic sequences of equal length corresponding to the shape, and repeat the operation above.

4) Finally, we get the sequence similarity table and diagrams as follows.

3.2 Model validation and evaluation

3.2.1 Model validation

1) Result Analysis

We used our model to analyze three sets of genetic sequences, each containing sequences from four or six species corresponding to a common trait. Each group takes the first sequence and compares it with itself and other sequences to obtain the Needleman-Wunsch similarity. The results are as follows:

Table1: Cell mitochondria-related apoptosis factors

Species	Base Sequence	Similarity
Human	MGKI(length: 274)	1.000000
Dog	MFRC(length: 272)	0.332130
Turtle	MFRC(length: 277)	0.296029
Rat	MFRC(length: 257)	0.328520

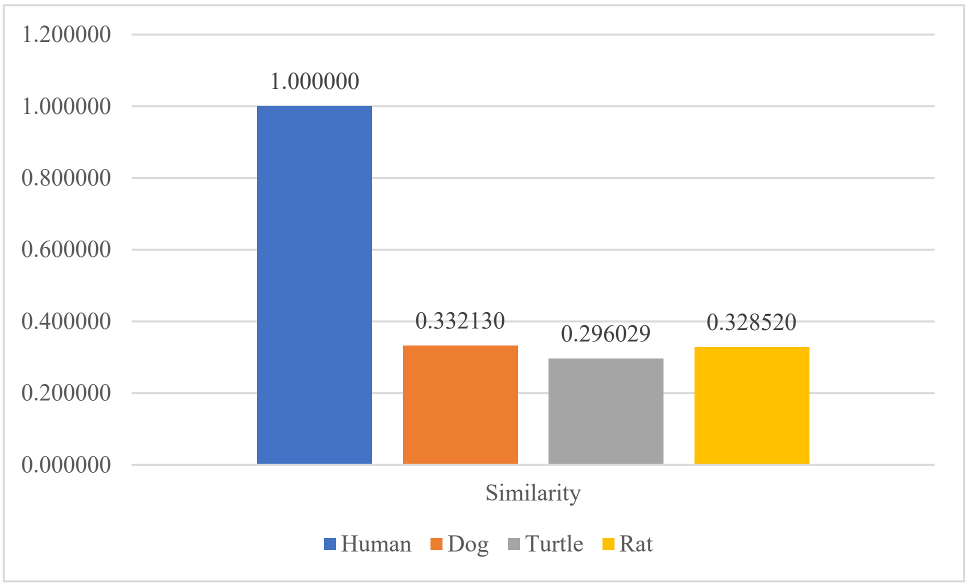

Fig1:Similarity between cell mitochondria-related apoptosis factors

Table2: Mitochondrial intima protein

Species	Base Sequence	Similarity
Northern fur seal	TCCC(length: 216)	1.000000
Chimpanzee	AAAG(length: 211)	0.542857
Rhesus monkey	GAAC(length: 212)	0.566667
Norwegian rat	GCCT(length: 220)	0.600000
Japanese quail	ATGT(length: 240)	0.619048
Homo sapiens	ACAC(length: 219)	0.642857

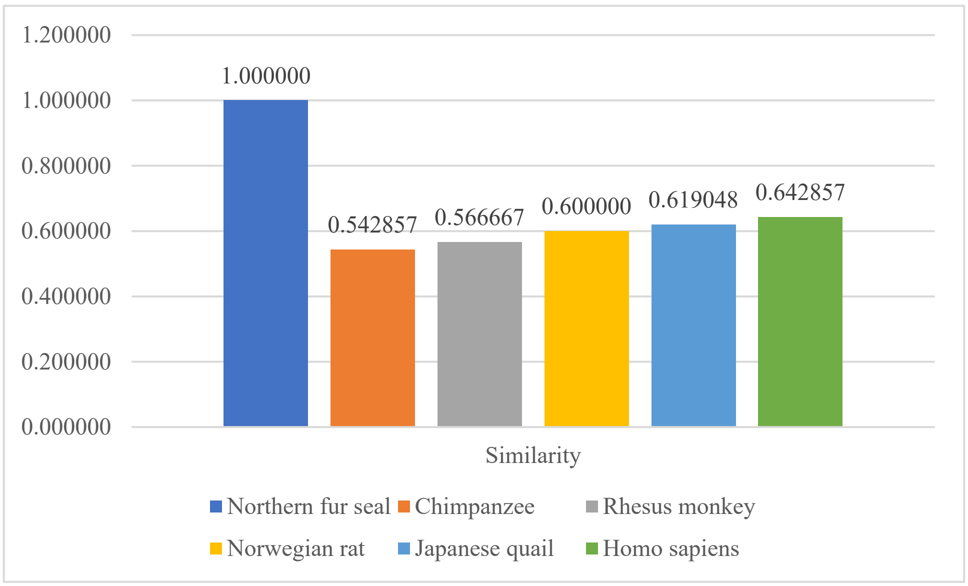

Fig2: Similarity between mitochondrial intima protein

Table3: Chloroplast heat shock protein

Species	Base Sequence	Similarity
Cucumber	GGAA(length: 217)	1.000000
Plum	GGAG(length: 220)	0.623810
Arabidopsis	GTGGlength: 219)	0.623810
Micromonas	ATGG(length: 229)	0.528571
Wheat	AGAA(length: 225)	0.614286
Osteococcus fluorescens	GTCT(length: 226)	0.595238

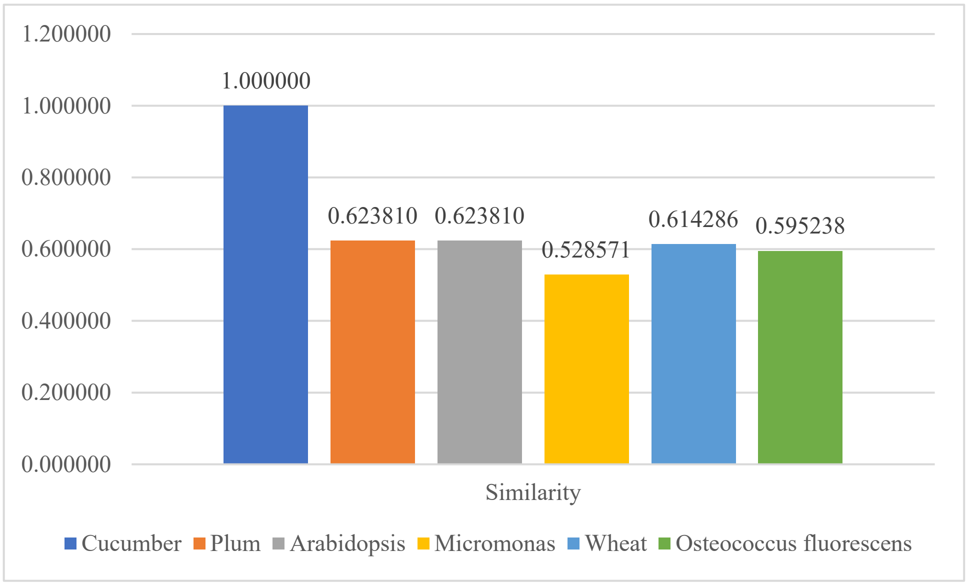

Fig3: Similarity between chloroplast heat shock protein

The obtained similarity is in line with our empirical judgment and expected results. Additionally, it is close to the given sequence similarity in the database.

2) Further verification

We use our model to Using this model to analyze the genetic sequence corresponding to apolipoprotein O in a total of 3 generations of members of a human family. We take the first sequence and compare it with itself and other sequences to calculate the Needleman-Wunsch similarity. The results are as follows:

Table4: Family apolipoprotein

Species	Base Sequence	Similarity

Parental generation P1	MFKV(length: 201)	1.000000
Filial generation F1	MVRL(length: 210)	0.989899
Filial generation F2	MVRL(length: 210)	0.989899
Filial generation F3	MFKL(length: 170)	0.818182
Parental generation P2	MFKV(length: 206)	1.000000
Filial generation F3	MFKL(length: 170)	0.818182

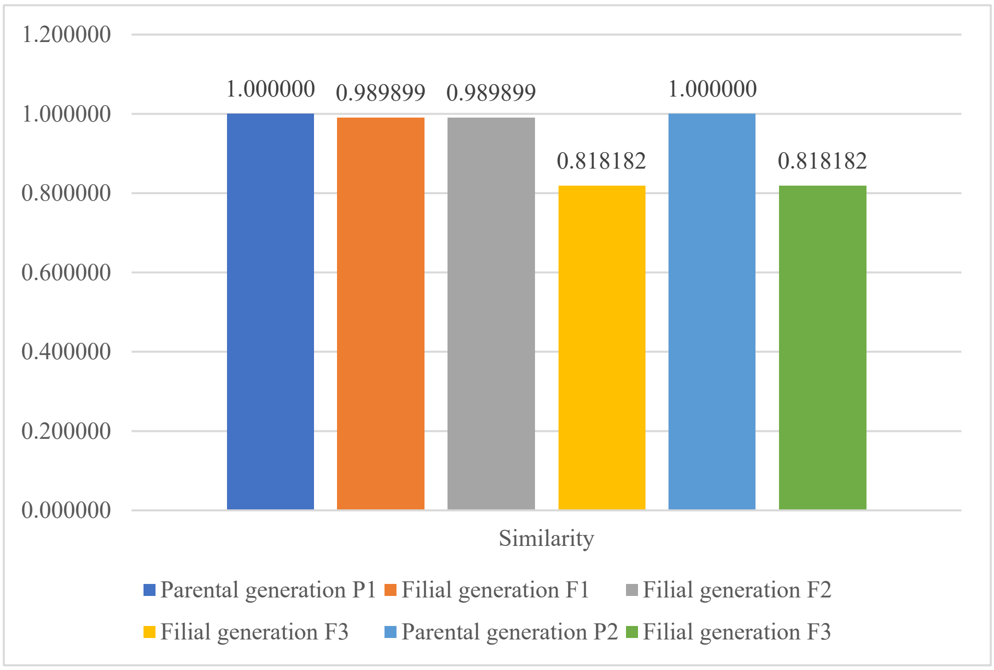

Fig4: Similarity between family apolipoprotein

Before data processing, we speculated that the obtained similarity is relatively high. Results illustrate that the obtained results are further in line with empirical judgment as well as expected results, and are close to the given sequence similarity in the database.

3.2.2 *Model Evaluation*

1) Complexity Analysis

In the calculation process, we use two strings to generate a scoring matrix according to the length of the two strings and perform dynamic programming, then backtrack to obtain the corresponding relationship of the sequences. The Needleman-Wunsch similarity can be inferred from this corresponding relationship. Considering the number of loops in the program, we believe that the time complexity of the algorithm applied by this model is $O(mn)$, where m, n correspond to the lengths of the two strings.

In the face of longer sequences and multiple sequence processing, the time consumption of our model will increase.

2) Strength and weakness

- Strength: The dynamic programming algorithm is introduced into the processing of biological sequences, which can measure the global similarity of sequences. Our model makes full use of known information in the sequences. It conforms to general cognition, and has versatility when dealing with genetic information that can be expressed using strings.
- Weakness: Our model requires high space and time complexity, and can only compare the similarity between two sequences. Therefore, it is relatively simple to deal with collections containing multiple sequences.

3.3 Improved Model

3.3.1 Extra Definitions

- **Average Needleman-Wunsch similarity**

Set the string s_0 and the string set s, s contains n strings, s_i is the i-th string in s, q_i is the Needleman-Wunsch similarity between s_0 and s_i. $\frac{1}{n}\sum_{i=1}^{n} q_i$ is the average Needleman-Wunsch similarity between s_0 and s, which is referred to $\bar{q}$.

3.3.2 Extra Assumptions

- The mutation probability of direct genetic factors in the genetic sequence is very low, and it is a small probability event. Therefore, for the sequence corresponding to a specific genetic trait, if the genetic sequences of the first generation offspring are all evolved from a specific ancestral genetic sequence, the similarity between them is high, and only a few sequences have significant genetic factor mutations.
- For the genetic sequences of multiple offspring, take a certain sequence and compare it with other sequences. Then the average Needleman-Wunsch similarity obtained can explain the overall similarity between this sequence and the rest of the sequences.
- In the genetic process, we believe that the lower the similarity, the greater the genetic algebra between two individuals in the genetic tree.
- For the strings corresponding to two sequences, after aligning them and dividing them into shorter substrings with appropriate length, their Needleman-Wunsch similarity is still valid.
- The process of calculating Needleman-Wunsch similarity after segmentation is

assumed to be the same as our basic model.

3.3.3 The Foundation of Model

On the basis of additional assumptions above, we propose a multi-sequence analysis model based on the average Needleman-Wunsch similarity of family genetic sequence sets to infer common ancestral sequences and obtain genetic trees.

- Based on the basic model, for a given family genetic sequence data set s, headers of each sequence s_i are aligned.
- According to the total number of sequences n in s, divide s_i $(i = 1,2,\dots n)$ into n parts, then take the first part of $s_1, s_2, \dots, s_n$to build sets $S_1, S_2, \dots, S_n$.
- Calculate the average Needleman-Wunsch similarity between the sequences s_{1i} $(i = 1,2,\dots n)$ in S_1 and the rest of the sequences, which is referred to as $\overline{q_{1i}}$.
- Compare $q_{11}, q_{12}, \dots, q_{1n}$, and take the subsequence s_{1i} corresponding to the maximum value as the first segment of the common ancestral sequence.
- Repeat steps 3 and 4 for $s_1, s_2, \dots, s_n$ to obtain the $2nd, 3rd, \dots, nth$ segments of the common ancestral sequence, then integrate the common ancestral sequence s_0.
- Compare s_0 with $s_1, s_2, \dots, s_n$ to get $q_1, q_2, \dots, q_n$.
- Compare $q_1, q_2, \dots, q_n$. According to our assumption, we believe that q_i $(i = 1,2,\dots n)$ can be divided into multiple levels according to the size. If the q_i values at the same level are close, it can be regarded as a genetic sequence separated from the ancestor sequence by the same generation.
- For sequences at different levels, repeat step 7 to draw a genetic tree.

3.3.4 Solutions and Results

- Solutions:

1) Based on the basic model, we write a program that can be used to read family sequences and conform to the extended model.
2) Through genetic databases (such as NCBI), we find several genetic sequences (length longer than 103 bases) corresponding to some common dominant traits of different species. Then, we use the xlwings library of Python to read the data, and process the program with extended model. According to the analysis above, we can derive the shared ancestral genetic sequence and draw the genetic tree.
3) According to the processing results, we draw tables and diagrams as follows.

Table5: Sequence Segmentation

Species	Base Sequence	Similarity
A	TTCT(length: 139)	1.000000
B	TTCT(length: 132)	0.989899
C	TTCT(length: 140)	0.989899
D	TTCT(length: 144)	0.818182
E	TTCT(length: 152)	1.000000
F(ancestral sequence)	TTCT(length: 136)	0.818182

4) Analyze original data of the data pool to determine the reliability of our improved model.

- Results:

For the genetic sequence of hepatic lipase (a process that affects receptor-mediated lipoprotein uptake in individuals) in a human family genetic sequence set containing 6 individual data (numbered A-F), we respectively start from the 497th position Click to intercept a fragment of an independently expressed protein (the default length is 140, the length may vary due to mutations) for analysis. After each sequence s_n is divided into 6 segments and processed, results show that:

Table6: Sequence Segmentation

	s11	s12	s13	s14	s15	s16
s11		0.956522	1.000000	0.913043	1.000000	1.000000
s12	0.956522		0.956522	0.869565	0.956522	0.956522
s13	1.000000	0.956522		0.913043	1.000000	1.000000
s14	0.913043	0.869565	0.913043		0.913043	0.913043
s15	1.000000	0.956522	1.000000	0.913043		1.000000
s16	1.000000	0.956522	1.000000	0.913043	1.000000	
Average	0.973913	0.939130	0.973913	0.904348	0.973913	0.973913
Selected sequence	s11					
Note: Data on the diagnal is empty since it cannot compare with itself						

Table7: Sequence Segmentation

	s_{11}	s_{12}	s_{13}	s_{14}	s_{15}	s_{16}
s_{11}		0.956522	1.000000	0.913043	1.000000	1.000000
s_{12}	0.956522		0.956522	0.869565	0.956522	0.956522
s_{13}	1.000000	0.956522		0.913043	1.000000	1.000000
s_{14}	0.913043	0.869565	0.913043		0.913043	0.913043
s_{15}	1.000000	0.956522	1.000000	0.913043		1.000000
s_{16}	1.000000	0.956522	1.000000	0.913043	1.000000	
$\bar{q}$	0.973913	0.939130	0.973913	0.904348	0.973913	0.973913
selected sequence	s_{11}					
(Note: Data on the diagnal is empty since it cannot compare with itself)						

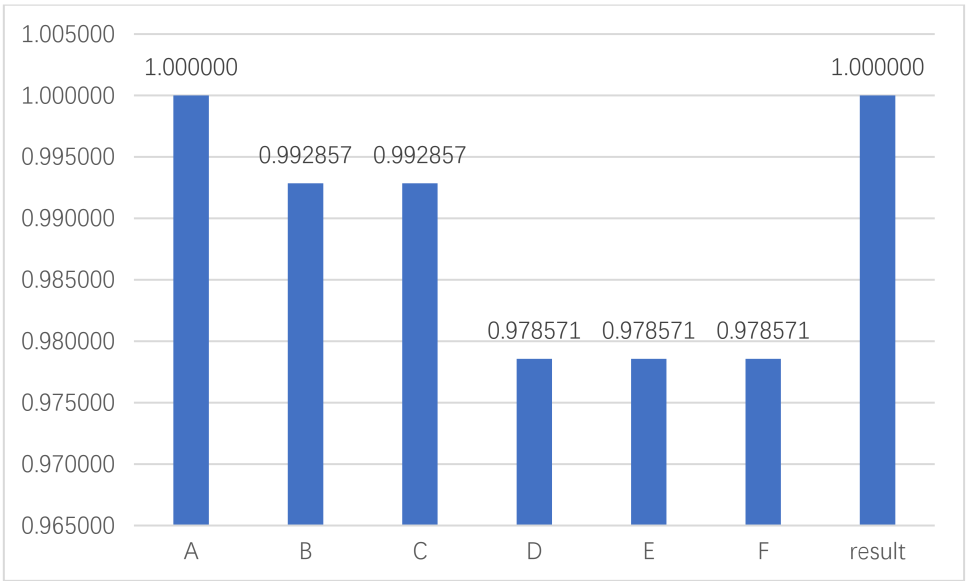

Fig5: Similarity between sequence segmentations

The spliced sequence is used as the common ancestral genetic sequence. Compared with the previous sequence, the 6 sequences can be divided into 3 levels:

- Level 1: Parental generation (sequence 1)
- Level 2: Filial Generation1 (Sequence 2, 3)
- Level 3: Filial Generation2 (Sequence 4, 5, 6)

For the Filial Generation1 and Filial Generation2, we use the above model to calculate the Needleman-Wunsch similarity. After dividing them into different levels, we draw the genetic tree as shown in the figure:

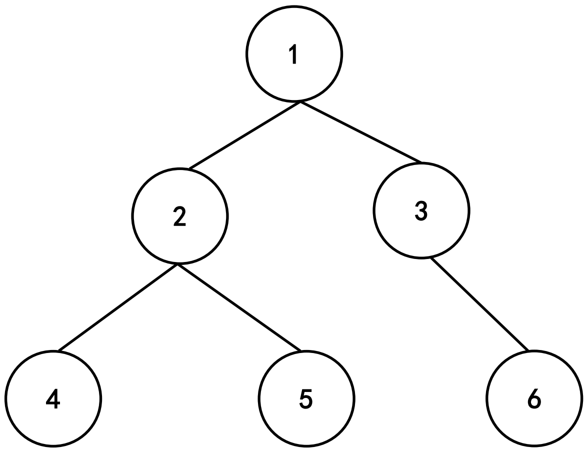

Fig6.Schematic diagram of genetic tree

3.3.5 Analysis of the Result

In this family, sequence 1 itself is the common ancestral sequence of other sequences. Additionally, the genetic tree is in line with empirical judgement and expected results. Verified with the multi-sequence analysis tools in the NCBI database, our conclusions are roughly consistent with the given information in the data pool. Therefore, we believe that this model is valid and reliable for the homology analysis results of family sequences.

3.3.6 Strength and Weakness

- Strength: We make full use of all the sequence information, and the obtained common ancestry sequence is highly convincing. Also, the parent-offspring relationship obtained when drawing the genetic information tree is relatively reliable. Even more importantly, the number of long sequence divisions determined according to the total number of multiple sequences makes the model applicable when faced with a large number of family genetic sequences.

- Weakness: The algorithm used by the model has high time and space complexity. Due to the random mutation of genetic base in a single genetic sequence, if the sequence of a progeny is mutated in the subsequent genetic process and has a high similarity to a genetic sequence in its parent, it may it may cause its position to shift upwards in the genetic tree.

IV. Conclusions

4.1 Conclusions of the problem

Our model contains:

- A genetic sequence homology analysis model based on dynamic programming.
- Method for proving the accuracy and validation of quantitative measurement of genetic sequence similarity.
- A model for inferring consensus ancestral sequences in family genetic sequence sets.
- Verification of the family genetic sequences to prove the accuracy and validation of the model.

4.2 Methods used in our models

- Dynamic programming
- Edit distance
- Needleman-Wunsch algorithm
- Python tools(xlwings)
- Excel and drawing tools

4.3 Applications of our models

- Our model can be applied to calculate the similarity of two sequences.
- For multiple genetic sequences, our model determines their ancestral sequence through sequence alignment.
- The genetic trees drawn from our models can be applied in species research or traits identification.

V. Future Work

5.1 Model Improvement

Due to time constraints, we were unable to include every variable in our model that we considered. As such, though we feel our model has successfully achieved calculating similarity and finding ancestral sequences, we feel that there is more work to be done to expand this into a more robust model.

We would like to address several key points in our future work. First, our model requires high space and time complexity, so it takes time to work out the genetic correlations between sequences. Our future work contains optimizing the time and space complexity of the program, such as reducing the maintenance times of variables in redundant state transition equations. If we further optimize our program, we can reduce the memory occupied by the program.

Furthermore, reducing accidental error can be achieved in our future work. To deal with the extremely accidental phenomenon that genetic base may restore after mutation in the family genetic sequence, we can preprocess the sequence. It is an effective way to reduce accidental errors by using null character padding to unify the sequence length, and then using the maximum common substring principle and the Jaro-Winkler edit distance algorithm (a more accurate algorithm for analyzing the complexity of short sequences) for further analysis.

Finally, to improve accuracy of models used in a large scale, RNN neural network can be applied to analyze the same traits among a large number of different species or the same traits among different individuals in the same family. This allows the predicted sequence to be obtained and compared to the actual sequence to improve accuracy of results.

5.2 Data Extension

Because of time constraints and hardware platform limitations, we only referenced part of the genetic sequence data to verify model validation. In bioinformatics engineering analysis, the amount and scale of data are often even larger, so any future expansion of our work should strive to expand the data pool. In order to expand the scope of application of the model and improve its accuracy, we can combine the algorithm improvement in 5.1, use genetic databases (such as NCBI) to obtain more genetic sequences, and use platforms with higher computing power for data analysis, so as to give more exact model.

VI. References

[1] 甘秋云.基于动态规划的Needleman-Wunsch双序列比对算法的分析与研究[J]. 计算机工程与科学,2021,43(02):340-346.

[2] A. E. E. -D. Rashed, H. M. Amer, M. El-Seddek and H. E. -D. Moustafa, "Sequence Alignment Using Machine Learning-Based Needleman–Wunsch Algorithm," in IEEE Access, vol. 9, pp. 109522-109535, 2021, doi: 10.1109/ACCESS.2021.3100408.

[3] Xiantao Jiang, Xueliang Fu, Gaifang Dong, Honghui Li. Research on Pairwise Sequence Alignment Needleman-Wunsch Algorithm[C]//.Proceedings of 2017 5th International Conference on Mechatronics, Materials, Chemistry and Computer Engineering(ICMMCCE 2017).,2017:1061-1066.

[4] S. El-Metwally, O. M. Ouda, and M. Helmy, Next Generation Sequencing Technologies and Challenges in Sequence Assembly, 2014, vol. 7, doi: 10.1007/978-1-4939-0715-1.

[5] 姜鲜桃. 双序列比对Needleman-Wunsch算法研究[D].内蒙古农业大学,2017.

[6] 李研. 生物序列比对算法的并行优化设计与实现[D].哈尔滨工业大学,2015.

[7] Hobbit, 编辑距离讲解 [EB/OL].(2020.04.09)[2022.12.10].
http://old.moe.gov.cn//publicfiles/business/htmlfiles/moe/s3342/201203/133322.html.

[8] 我不是大熊,用 Python 从头实现 Needleman-Wunsch 序列比对算法 [EB/OL].(2019.11.29)[2022.12.10].https://zhuanlan.zhihu.com/p/78027062?utm_id=0.

VII. Appendix

Programs and code

nwdistance.py(used to calculate nwdistance and similarity)

// Code block

```python
def theta(a, b):
    if a == '-' or b == '-' or a != b:  # gap or mismatch
        return -1
    elif a == b:  # match
        return 1


def make_score_matrix(seq1, seq2):
    """
    return score matrix and map(each score from which direction)
    0: diagnosis
    1: up
    2: left
    """
    seq1 = '-' + seq1
    seq2 = '-' + seq2
    score_mat = {}
    trace_mat = {}

    for i, p in enumerate(seq1):
        score_mat[i] = {}
        trace_mat[i] = {}
        for j, q in enumerate(seq2):
            if i == 0:  # first row, gap in seq1
                score_mat[i][j] = -j
                trace_mat[i][j] = 1
                continue
            if j == 0:  # first column, gap in seq2
                score_mat[i][j] = -i
                trace_mat[i][j] = 2
                continue
            ul = score_mat[i - 1][j - 1] + theta(p, q)  # from up-left,
mark 0
            l = score_mat[i][j - 1] + theta('-', q)  # from left, mark
1, gap in seq1
            u = score_mat[i - 1][j] + theta(p, '-')  # from up, mark 2,
gap in seq2
            picked = max([ul, l, u])
            score_mat[i][j] = picked
            trace_mat[i][j] = [ul, l, u].index(picked)  # record which
direction
    return score_mat, trace_mat
```

main.py (working with the data using the basic model)
// Code block

```
import xlwings as xw
import nwdistance as nw

wb = xw.Book("./dataset/家族载脂蛋白 O.xlsx")
sht = wb.sheets["Sheet1"]
for i in range(1, 7):
    print(sht.range("A" + str(i)).value)
    q = nw.nwdistance(sht.range("B" + str(i)).value,
sht.range("B1").value) / len(sht.range("B1").value)
    print(q)
    sht.range('C' + str(i)).value = q
```

family.py (Processing data using the extension model)
// Code block

```
import nwdistance as nw
import xlwings as xw


def numbertoalphabet(i):
    letter = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K',
'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U',
              'V', 'W', 'X', 'Y', 'Z']
    return letter[i]


wb = xw.Book("./dataset/family.xlsx")
sht1 = wb.sheets["Sheet1"]
sht2 = wb.sheets["Sheet2"]

sht = [wb.sheets["Sheet3"],
      wb.sheets["Sheet4"],
      wb.sheets["Sheet5"],
      wb.sheets["Sheet6"],
      wb.sheets["Sheet7"],
      wb.sheets["Sheet8"],
      ]


for h in range(1, 7):
    for i in range(1, 7):
        for j in range(1, 7):
            maxlen = max(len(sht2.range(numbertoalphabet(i - 1) +
str(h)).value),
                           len(sht2.range(numbertoalphabet(j - 1) +
str(h)).value))
            q = nw.nwdistance(sht2.range(numbertoalphabet(i - 1) +
str(h)).value,
                               sht2.range(numbertoalphabet(j - 1) +
str(h)).value) / maxlen
            sht[h - 1].range(numbertoalphabet(i) + str(j + 1)).value =
q

for i in range(1,7):
    print(sht1.range("A" + str(i)).value)
    q = nw.nwdistance(sht1.range("B" + str(i)).value,
sht1.range("B1").value) / len(sht1.range("B7").value)
    print(q)
    sht1.range('C' + str(i)).value = q
```

See the attachment for the source code file
Data set
Too large to be shown in the paper, see attachment.