Research on global double sequence alignment optimization algorithm based on dynamic programming

Yuanzi He*
Computer College, Guangdong University of Science and Technology, Dongguan, Guangdong, China

ABSTRACT

Biological sequence alignment is one of the most basic research topics in bioinformatics. The double sequence alignment algorithm based on dynamic programming mainly uses iterative algorithm and vacancy penalty rules to compare gene sequences one by one, calculate their similarity scores, and finally obtain the best alignment between sequences through backtracking analysis. This paper studies the analysis and implementation of the global double sequence alignment optimization algorithm based on dynamic programming.

Keywords: Bioinformatics, sequence global alignment, dynamic planning, replacement matrix

1. INTRODUCTION

Bioinformatics is a new subject formed by the intersection of biology, computer science and applied mathematics. Sequence alignment is the core of biological computing and the most basic and important method in biology[1]. It provides a powerful way to try to indicate whether there is enough similarity between two sequences. The most common alignment is the pairwise alignment between protein sequences or nucleic acid sequences. By comparing the similarity regions between the two sequences, we can find the possible molecular evolution relationship between them. The classification of sequence alignment is divided into pair wise sequence alignment and multiple sequence alignment in terms of the number of sequences that are aligned at the same time; Considering the comparison scope, it can be divided into global alignment and local alignment[2].

2. DYNAMIC PROGRAMMING ALGORITHM IDEA

Double sequence alignment is one of the common methods of sequence analysis[3], and it is also the basis of multi sequence alignment and database search. Therefore, there are many mathematical methods to solve the problem of double sequence alignment, among which the most basic, effective and widely recognized method is dynamic programming[4]. Most sequence alignments, such as global alignment, local alignment and hidden Markov model, take dynamic programming algorithm as the core algorithm of vacancy insertion[5].

The dynamic programming method is limited in practical use because of its huge time and space complexity. Therefore, for sequence alignment algorithm, the current key problem is how to reduce the running time and space of the algorithm on the basis of maintaining biological sensitivity[6].

The basic idea of dynamic programming algorithm to solve the problem of sequence alignment can be described as: using the iterative method to calculate the similar scores of two sequences and store them in a score matrix. According to this score matrix, the optimal alignment sequence is searched back[7].

The elements of the score matrix are iteratively calculated by the following formula:

$$\begin{aligned} M_{i,j} = \ & max\{M_{i-1,j-1} + M(s_i, t_j), \\ & max(M_{i-x,j} - W_x), \\ & max(M_{i,j-y} - W_y)\} \end{aligned} \tag{1}$$

Matrix elements (i, j) can be reached from three directions: diagonal elements, elements in the same row or column, see Figure 1 for details.

* 404250469@qq.com

Third International Conference on Computer Science and Communication Technology (ICCSCT 2022)
edited by Yingfa Lu, Changbo Cheng, Proc. of SPIE Vol. 12506, 125060L
 · 0277-786X · doi: 10.1117/12.2662630

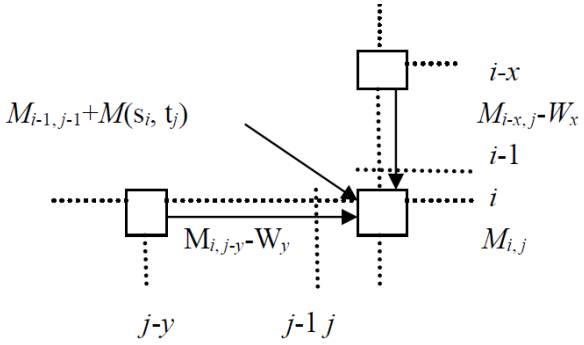

Figure 1. Three sources of matrix elements.

In the score matrix, there are three possible paths to an element with positions i and j: Through the diagonal direction of bits i-1 and j-1, there is no vacancy penalty; Through the vertical direction of column j and the horizontal direction of row i, the value of vacancy penalty depends on the number of inserted vacancies. If the two sequences are $S = s1, s2, \ldots, sm$ and $T = t1, t2, \ldots, tn$, $Mi, j=M(s1s2\ldots$ sm, $t1t2\ldots tn)$, Mi, j is the comparison score between the ith character in the arrival sequence s and the jth character in the sequence T, $M(si, tj)$ is the score value of si and tj, Wx is the vacancy penalty value with the vacancy length of x on the sequence s, and Wy is the vacancy penalty value with the vacancy length of y on the sequence T. Here, Mi, j is the score value of the best comparison path obtained from three directions in the score matrix. When all elements Mi, j of the score matrix are calculated, the end point of the best path is at the position of the last row and the last column. Then from this point on, according to the above formula, the path found by backtracking in the score matrix is an optimal path.

3. DYNAMIC PROGRAMMING TO ACHIEVE GLOBAL DOUBLE SEQUENCE ALIGNMENT OPTIMIZATION ALGORITHM ANALYSIS

Dynamic programming realizes the global double sequence alignment optimization algorithm, which introduces the vacancy penalty mechanism[8]. This algorithm also uses a matrix, which will input two sequences, one along the top and one along the left. It can also reach each cell through the following three ways[9]:

- From the cell above, it means to compare the character on the left with the space.
- The cell from the left represents the comparison of the above characters with the empty bits.
- The cell from the upper left represents the comparison with the characters on the left and above.

The complete table is given below. You can go back and check it when explaining how to fill in the table, as shown in Figure 2:

		G	C	C	C	T	A	G	C	G
	0	-2	-4	-6	-8	-10	-12	-14	-16	-18
G	-2	1	-1	-3	-5	-7	-9	-11	-13	-15
C	-4	-1	2	0	-2	-4	-6	-8	-10	-12
G	-6	-3	0	1	-1	-3	-5	-5	-7	-9
C	-8	-5	-2	1	2	0	-2	-4	-4	-6
A	-10	-7	-4	-1	0	1	1	-1	-3	-5
A	-12	-9	-6	-3	-2	-1	2	0	-2	-4
T	-14	-11	-8	-5	-4	-1	0	1	-1	-3
G	-16	-13	-10	-7	-6	-3	-2	1	0	0

Figure 2. Optimized algorithm score matrix.

First of all, the table must be initialized, which means that the scores in the second row and the second column and the arrows representing the source are filled. The operation of filling the second row means that the characters in the first sequence at the top are used and the spaces are used instead of the first characters in the sequence from top to bottom on the left. The bonus of the spaces is -2, so every time the spaces are used, the previous cells are added -2 points. The previous cell is the cell on the left, which explains why we get 0, -2, -4, -6 In such a sequence, the scores and arrows in the second column are obtained in a similar way, as shown in Figure 2.

Secondly, you need to fill in the remaining cells. For each cell, there are three choices. To choose the largest one, you can reach each cell from the top, left and top left. False S and T are the strings to be compared, S" and T" are the strings in the generated comparison. Reaching the cell from above is equivalent to adding the left character from T to T", skipping the current character in S above and adding a space in S". Because the score of a vacancy is -2, the score of the current cell is obtained by subtracting 2 from the score of the upper cell. Similarly, by subtracting the score of the left cell by 2, you can reach the empty cell from the left. Finally, you can add the above characters to S" and the left characters to T". This is equivalent to entering a blank cell from the top left. These two characters will match. In this case, the new score is the score of the upper left cell minus 1. Among the three possibilities, the one with the largest score is chosen.

Finally, you need to get the actual comparison string S" and T" and the comparison score. The score in the lower right cell contains the maximum comparison score of S and t. to get S" and T", you need to start from the lower right cell and trace back along the arrow to build S" and T" in reverse. It can be seen from the construction process of the table that from top to bottom, the left character is added from t to T" and the vacancy is added to S"; From left to right, it corresponds to adding the above characters from s to S" and adding empty bits to T"; Moving down and right means adding characters from s and t to S" and T", respectively.

4. IMPLEMENTATION OF OPTIMIZATION ALGORITHM COMPARISON

4.1 Input two sequences

The user is prompted to enter the two sequences to compare, which can be nucleotide sequences or protein sequences. A variety of input methods are provided for users to choose.

- The users can directly input or copy and paste;
- The users can read a single sequence file;
- The users can read double sequence file;
- Nucleotide sequences or protein sequences are randomly generated.

The files here can be text files (.Txt) or fasta files (.fasta). For other format sequence files, you can use Clusta1x software to convert the format. You can enter the sequence GCCCTAGCG and the target sequence CGCCAATG to query.

4.2 Setting of scoring parameters

System scoring parameter is set as follows: when selecting vacancy parameters, it is largely empirical, and the selected score will rarely have theoretical support. If the value of vacancy penalty is too high, there may be no vacancy in the comparison process; If it is too low, too many vacancies will be introduced, so users can set the parameters of the vacancy penalty model according to actual needs.

Here, the set parameters are: match = 1.0, mismatch = -1.0, gap = -2.0.

4.3 Build score matrix

It mainly initializes the score matrix. The comparison sequence characters are filled into the matrix, and the values of the matrix cells in row 0 and column 0 are initialized.

4.4 Output comparison results

The comparison results show the length of the two sequences, the number of characters matched, the number of characters mismatched, the similarity, the number of Gaps, scores, and time consumption. The comparison results are shown in Figure 3:

```
G C C C T A G C G   9
| | * |   | * * |
G C G C - A A T G   8
Length:          9
match:           5/9        55.6%
mismatch:        3/9        33.3%
gap:             1/9        11.1%
score:           0.0
```

Figure 3. Comparison results.

In the figure, | represents match, * represents mismatch, and - represents Gap.

At the same time, the scoring matrix table is also displayed. The lower right corner of the scoring matrix is the final score. Now let's check the calculation results. There are five matching points, one vacancy and three dislocations. Score: $1\times5+(-2)\times1+(-1)\times3=0$, which is the best global sequence alignment, but it is not unique.

		G	C	C	C	T	A	G	C	G
	0	← -2.0	← -4.0	← -6.0	← -8.0	← -10.0	← -12.0	← -14.0	← -16.0	← -18.0
G	↑ -2.0	↖ 1.0	← -1.0	← -3.0	← -5.0	← -7.0	← -9.0	↖ -11.0	← -13.0	↖ -15.0
C	↑ -4.0	↑ -1.0	↖ 2.0	↖ 0.0	↖ -2.0	← -4.0	← -6.0	← -8.0	↖ -10.0	← -12.0
G	↑ -6.0	↖ -3.0	↑ 0.0	↖ 1.0	↖ -1.0	↖ -3.0	↖ -5.0	↖ -5.0	← -7.0	↖ -9.0
C	↑ -8.0	↑ -5.0	↖ -2.0	↖ 1.0	↖ 2.0	← 0.0	← -2.0	← -4.0	↖ -4.0	← -6.0
A	↑ -10.0	↑ -7.0	↑ -4.0	↑ -1.0	↖ 0.0	↖ 1.0	↖ 1.0	← -1.0	← -3.0	↖ -5.0
A	↑ -12.0	↑ -9.0	↑ -6.0	↑ -3.0	↖ -2.0	↖ -1.0	↖ 2.0	↖ 0.0	↖ -2.0	↖ -4.0
T	↑ -14.0	↑ -11.0	↑ -8.0	↑ -5.0	↖ -4.0	↖ -1.0	↑ 0.0	↖ 1.0	↖ -1.0	↖ -3.0
G	↑ -16.0	↖ -13.0	↑ -10.0	↑ -7.0	↖ -6.0	↑ -3.0	↖ -2.0	↖ 1.0	↖ 0.0	↖ 0.0

Figure 4. Scoring matrix scoring matrix.

The arrow in figure 4 represents the source of the current cell, and the Yellow cell represents the backtracking path. There may be multiple backtracking paths. Considering that the user graphical interface cannot display multiple paths, this algorithm only calculates an optimal path.

LCS algorithm does not introduce the vacancy penalty mechanism. By comparing the length of the longest common subsequence of the two sequences, it reveals the similarity of the sequences. The longer the common subsequence, the more similar the two sequences are. The comparison results are as follows:

```
G C C C T A G C T
| | * |   | * * |
G C G C - A A T G
```

So the LCS is GCCAG, the longest common subsequence, as shown in Figure 5:

		G	C	C	C	T	A	G	C	G
	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
G	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
C	0.0	1.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
G	0.0	1.0	2.0	2.0	2.0	2.0	2.0	3.0	3.0	3.0
C	0.0	1.0	2.0	3.0	3.0	3.0	3.0	3.0	4.0	4.0
A	0.0	1.0	2.0	3.0	3.0	3.0	4.0	4.0	4.0	4.0
A	0.0	1.0	2.0	3.0	3.0	3.0	4.0	4.0	4.0	4.0
T	0.0	1.0	2.0	3.0	3.0	4.0	4.0	4.0	4.0	4.0
G	0.0	1.0	2.0	3.0	3.0	4.0	4.0	5.0	5.0	5.0

Figure 5. LCS algorithm results.

5. CONCLUSION

By comparing the similar regions and conservative sites between the two sequences, we can find the possible molecular evolution relationship between them. At present, there are many mathematical methods to solve the problem of double sequence alignment. Among them, the most basic, effective and widely recognized method is the dynamic programming method, which is implemented by using the dynamic scripting language Ruby and C++ language.

ACKNOWLEDGMENTS

2019 Guangdong Higher Education Teaching Reform Project—Exploration of embedded system curriculum reform under CDIO mode. Supported by Natural Science Project of Guangdong University of Science and Technology (GKY-2021KYZDK-6).

REFERENCES

[1] Shi, H. H. and Zhou, W. X., “Design and implementation of double sequence alignment algorithm component based on dynamic programming,” Computer Research and Development 56(09), 1907-1917 (2019).
[2] Gan, Q. Y., “Analysis and research of Needleman Wunsch double sequence alignment algorithm based on dynamic programming,” Computer Engineering and Science 43(02), 340-346 (2021).
[3] Li, D., “Research and improvement of double sequence alignment algorithm,” Electronic Technology and Software Engineering (18), 148 (2107).
[4] Pan, D. and Zhong, C., “Long sequence alignment parallel algorithm for two-level hash global and local index filtering,” Small Microcomputer System 9, 1-8 (2021).
[5] Cao, L., Xu, Y. L. and Deng, C. B., “Algorithm of DNA double sequence alignment problem,” Computer System Application 24(09), 112-117 (2015).
[6] Li, C.,Research and Parallel Optimization of Double Sequence Alignment Algorithm, Xi’an University of Electronic Science and Technology, Master’s Thesis, (2011).
[7] Jiang, X. T.,Research on Dual Sequence Alignment Needleman Wunsch Algorithm, Inner Mongolia Agricultural University, Master’s Thesis, (2107).
[8] Jiao, Y., Gao, J. and Zhang, W. G., “Research progress of two sequence alignment algorithms and software,” Computer Applications and Software 32(6), 5-8 (2015).
[9] Xiong, T., “Research and optimization of DNA sequence matching algorithm,” Information and Computer (Theoretical Edition) (10), 30-31 (2019).