Advances in Computer Science Research, volume 50
4th International Conference on Electrical & Electronics Engineering and Computer Science (ICEEECS 2016)

The application of binary path matrix in backtracking of sequences alignment

Zhongxi Cai[1, 2, a], Chengzhen Xu[1, b], Ying Wang [1, 3, c], Wang Cong[1,*]

[1] College of Automation, Harbin Engineering University, 145 Nantong Street, Nangang District, Harbin, Heilongjiang, 150001, China

[2] Mudanjiang Normal University, Mudanjiang, 157011, China

[3] Qiqihar University, Qiqihar, 161006, China

[a] 19809346@qq.com [b]136927320@qq.com [c]wangying0129@126.com

* Correspondence author: cwcy9156@163.com

Keywords: binary path matrix; dynamic programming; sequence alignment; backtracking.

Abstract. The sequence alignment is an extremely useful tool for molecular and evolutionary biology, and several programs and algorithms have developed for this purpose. The dynamic programming still keeps the most important position due to its highest accuracy, however, the accuracy of the algorithm is obtained by inserting or deleting some bases at the expense of possibly changing the basic or primary information of the original bases, it will lead to large errors in the prediction of the structure and function. This paper proposed an optimization algorithm based on the backtracking of the binary path matrix which is suitable for both global double sequence alignment and local sequence alignment. A scoring matrix is constructed based on the dynamic programming algorithm and a binary path matrix which is constructed at the same time. Then, the backtracking paths are extracted from the path matrix based on backtracking algorithm, and each backtracking path represents a alignment results. In this algorithm, global and local sequence alignment can be conducted at the same time. Compared with other sequence alignment algorithms, our proposed method can find more bases which are the same ones and provide better basis for analyzing the similarity of sequences.

1. Introduction

Sequence alignment is a highly scrutinized and fundamental approach in several research domains in molecular biology and Bioinformatics. And it is also crucial and significant for the computational biology[1]. The aim of sequence alignment is to analyze the sequence similarities from the levels of nucleic acids and amino acids[2], and predict the relationship of their functions and evolution. The sequence alignment lays the foundation for gene identification[3], molecular evolution and research of life origin[4]. As well known, sequence alignment is also a strong NP-hard combinatorial optimization problem[5, 6].

The previous sequence alignment method is the double sequence alignment. In 1970, Needleman[7] and Wunsch[8] formulated the dynamic programming algorithm. Considering the similarity of the entire sequence rather than the local similarities of the sequences[1], an alignment method which is applied to global similarity alignment of two sequences reveals that some matching sequence segments may hide some local similarities[9]. At the same year, Gibbs and McIntyp[10] put forward the bitmap matrix[11]. In 1981, Smith and Waterman[12] improved the dynamic programming algorithm based on the global function, and it become one of the most optimal local algorithms, In this algorithm, the sequence segments with high local similarities from two biological sequences could be located. But with the expansion of the biological data, Smith-Waterman algorithm cannot meet the need for the research of bioinformatics. BLAST, put forward by Pearson and Lipman in 1985, FastA[7] and SFAltschul[4] in1990, are the most famous Database searching algorithms which improved the local similarity algorithms. However the results of FASTA are not the most optimal. And Blast algorithm has three important drawbacks due to its

huge databases: the slowness in retrieving rate, high requirements for the I/O system and the enormous storing space occupied by the program[13]. Most popular sequence alignment tools, such as Clustal[14], T-coffee[15-18], SAGA[19, 20], PHGA[21], HMMER[22-24] were presented.

Currently, the sequence alignment is increasing widely applied in bioinformatics[25], including the gene and protein sequence alignments[26] of the RNA sequence alignment[27, 28] and even to the prediction of the RNA substandard structures. Some intelligence algorithms have been applied to sequence alignment, such as adaptive algorithm, parallel hybrid algorithm, genetic algorithm. To improve the accuracy of the algorithm, Fernando Jose Mateus da Silva[29] suggested integrating the local optimal search into the genetic algorithm. But it is difficult to go beyond the dynamic programming algorithm. Meanwhile, among the well-used sequence alignments, the method of the dynamic programming still keeps the most important position due to its highest accuracy[30]. This relatively accurate result is obtained by inserting or deleting some bases at the expense of possibly changing the basic or primary information of the original bases, which may exist large errors in the prediction of the structure and function.

The proposed algorithm in this paper suppresses part of the weaknesses in the dynamic programming and original sequence alignments algorithms. In the proposed algorithm, a scoring matrix is constructed based on the dynamic programming algorithm, and a path matrix is constructed at the same time. Then, the backtracking paths are extracted from the path matrix based on backtracking algorithm[31], and each backtracking path represents a alignment results. The new method is not limited to a single path so that all paths and multiple results can be found, on which the results continue to be improved and filtrated in order to find the best alignment results.

2. Method

2.1. Calculating the scoring matrix

The key problems of the double sequence alignment in the dynamic programming are to calculate the scoring matrix and trackbacking the best sequence alignments. As a result, when the dynamic programming algorithm is used to solve the problems of the sequence alignments, it needs not only one scoring matrix but also one path matrix, and they are used to track back to construct the maximum scores and the densest matches.

Assuming that there are two sequences, one sequence $S=s_1,s_2,\ldots s_m$, whose length is designate m, and the other sequence $T=t_1, t_2,\ldots t_n$, whose length is designate n, by which one scoring matrix M and one path matrix can be defined. The elements $M(i, j)$ in the scoring matrix means that the *i-th* base in the sequence S and the *j-th* base in the sequence T can be analyzed against the previous ones to get the last result of the optimal alignment. The elements in the scoring sequences can be calculated by the *Formula (1)*, *Formula (2)* and *Formula (3)* respectively.

$$M(i,0)=\sum_{k=1}^{i} D(s(k),-), \quad 1\le i\le m \tag{1}$$

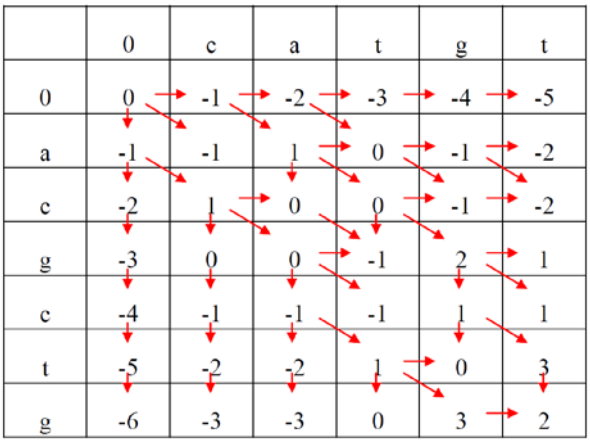
$$M(0,j)=\sum_{k=1}^{j} D(-,t(k)), \quad 1\le j\le n \tag{2}$$

$$M(i,j)=\max\begin{Bmatrix} M(i-1)+D(S(i),-) \\ M(i,j-1)+D(-,t(j)) \\ M(i-1,j-1)+D(s(i),t(j)) \end{Bmatrix} (1\le i\le m,\ 1\le j\le n) \tag{3}$$

The judgment element of the first line and the first column in the scoring matrix is not the first character of the sequence, defined as 0. $M(0, 0)=0$. While the first character of the two sequences become the second line or column, so that the first line of each sequence cannot lose its meaning of scoring. "-" indicates being inserted or deleted. D means the scoring function about one character, and $D(s[i], t[j])$ represents one score of the *i-th* character in S sequence against the *j-th* character in T sequence, while $D(s[i], -)$ and $D(-, [j])$ denote one score of the character against the blank, and also means the gap penalty.

For example, there are two random sequences, one sequence S=acgctg, and the other sequence T=catgt, in which, by definition, the blank penalties are $D(s[i], -)$ and $D(-, [j])=-1$, the same

alignment for $s[i]$ and $t[j]$ is $D(s[i], t[j])=2$, and the different one $D(s[i],t[j])=-1$.so the alignment scoring matrix of the sequence S and T from the *Formula (1)*, *Formula (2)* and *Formula (3)*,as shown in Fig.1.



	0	c	a	t	g	t
0	0	-1	-2	-3	-4	-5
a	-1	-1	1	0	-1	-2
c	-2	1	0	0	-1	-2
g	-3	0	0	-1	2	1
c	-4	-1	-1	-1	1	1
t	-5	-2	-2	1	0	3
g	-6	-3	-3	0	3	2

Fig. 1 the scoring matrix

In Fig.1, the arrows means that the value is calculated from the other end of the arrow, and it is obvious to observe the operational path of the whole matrix.

2.2. Establishing the Path Matrix

The backtracking from the highest score position to $M(1, 1)$ of the matrix is obtain by calculating the scoring matrix. Accordingly, a path matrix is established to find the same bases. In the process of obtaining the scoring matrix, the path of scoring matrix only need be store in our method. The path matrix $L(i, j)$ is calculated as shown in Fig.2, where $L(1, 1) = 0$.

2^0	2^1
2^2	L(i,j)

Fig.2. Calculation of path matrix

If $M(i, j)$ is calculated by $M(i\text{-}1, j\text{-}1)$, the two characters in this position are matched, and $L(i, j)= 2^0$; If $M(i, j)$ is calculated by $M(i, j\text{-}1)$ in the horizontal direction of the matrix, the $(i+1)$-*th* position of the sequence *s* needs to be insert using a blank space to match the *j-th* position in the sequence *t* and $L(i, j) = 2^1$; If $M(i, j)$ is calculated by $M(i\text{-}1, j)$ in the vertical direction of the matrix, then the $(j+1)$-*th* position in the sequence *t* needs to be inserted using a blank space to match the *j-th* of sequence *s* and $L(i, j) =2^0$; If multiple characters $M(i, j)$ which are calculated by different directions are equal, then we considered that $L(i, j)$ is calculated by *Formula (4)*. Taking *a*, *b*, *c* as the judgment basis, if $M(i, j)$ can be derived from corresponding direction, then it is set as 1, otherwise set as 0.

$$L(i, j) = a * 2^0 + b * 2^1 + c * 2^2 \tag{4}$$

Since each score can be known by path matrix, the backtracking paths are no redundancy; it is the fastest, and optimal. Fig.3 shows the path matrix.

2.3. The results of the backtracking search optimal alignment

The algorithm does not need global backtracking to search the optimization because of having path matrix. It is obvious where the score of matrix come from, therefore it can reduce the backtracking path, further to find the optimization alignment.

If the path matrix is used to search from back to front, there are lots of branch paths, therefore if the alignment sequence is longer; it will lead to too much branch paths. At this time, the backtracking can be started form the highest score position, and the same position of path matrix is used to finding the optimal path.

For example, the highest score of the alignment score matrix between sequence S and sequence T is 3, then the backtracking is upward from 3. According to the result of path matrix, the binary numeral in the position is compared with 1, 2, 4 respectively, if the result is not 0, then it is the node

of path. Similarly, the path is recorded by finding the circuit until the first position is reached. The arrows in figure indicate the backtracking path as shown in Fig.3.

	0	c	a	t	g	t
0	0	4	4	4	4	4
a	2	1	1	4	4	4
c	2	1	6	1	5	5
g	2	2	1	7	1	4
c	2	3	3	1	2	1
t	2	2	3	1	6	1
g	2	2	3	2	1	6

Fig. 3 The path matrix and backtracking

Through the process of backtracking path matrix, the alignment result is the record of backtracking process, and other values are overlooked and not backtracking, the process is described as Fig.4.

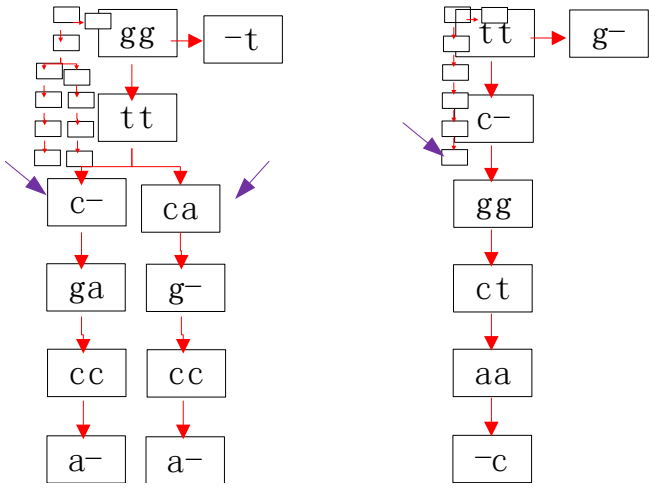


Fig. 4 Backtracking record

Above all, three alignment results are shown as:

Alignment result 1:
S: a c g c t g -
T: - c - a t g t

Alignment result 2:
S: a c g c t g -
T: - c a - t g t

Alignment result 3:
S: - a c g c t g
T: c a t g - t -

All the three results can be regarded as the optimal results. However, when the length of the sequence is longer, more branches will exist in backtracking course, so we do not know which one is the best alignment result, and which one is most suitable for the study of similarity. Therefore, we must continue to reduce the alignment results and it is better to find the best one.

2.4. Filtrating the alignment results

According Section *C*, there will be a variety of different results based on alignment algorithm. Therefore the sequence becomes longer and more. The results cannot be compared to determine the advantages without suitable filtrating conditions. Therefore, some conditions need to be added so that the best results can be displayed.

(1) Calculate the number of base pairs which are successfully matched in all the matching results. Name the *N-th* matching base pairs a_n, and then look for the biggest sequence or serial ones of a_n, list as follow:

$$a = max(a_n) \tag{5}$$

If a_n=a, the result is outputted, and the initial choice is completed.

(2) In this case, a lot of results are produced and the best matching result still can not be known. Thus, it is necessary to calculate each number of stem regions in matching results after first screening. A length of continuous successful matching is a stem region, and define the number of stem regions of the *n-th* matching results b_n , then

$$b = min(b_n) \tag{6}$$

If b_n = b, outputs the results, at this time the results are less or even only one.

(3)Less number of stem regions denotes that the same bases are concentrated in a few centralized places. It is obvious that there is a relation between two sequences or one is a variation from another. However, due to sequencing errors and problems of sequences restructuring there will be some insignificant bases influencing the judgment. At this point, some functions such as deleting or adding need to be added, which makes the alignments more readable. When a sequence has three or more identical bases at a certain position, while another sequence also has the same bases at the same position, however, the number of the two bases is different, e.g., Fig.5.

```
GUUGUAUAGUUUUAGGGUCACACCCACCA
|||||||||  |||||    |  |||   ||   |
GUUGUAUAG–UUUAGA AUUACA–UCA––A
```

Fig.5 Delete instances

There is -UUUU- in the first base, and -UUU- appears in another base at the same position resulting in a left U. It is defined that if the number is more than 3, deletion is primary. If the number is equal to or less than 3, it is priority to adding, while if more than 3, it is priority to deleting. Therefore, Fig.5 needs to delete a U in the first sequence. If all the addition and deletion are completed and the matching results are different, then repeat step 1 and 2. Finally, the optimal alignment results can be obtained.

3. Results

In the two same sequence alignments, the algorithm in this paper is more accurate than widely-used common algorithms such as *Needlman-Wunseh*, *Smith-Waterman* and *BLAST*, because it can identify the local similarities with great sensitivity. What's more, the algorithm can easily find the desirable results without ignoring or concealing the local similarities. Just as described in the following graph, to compare two sequences, *A*=AGTCTTGAGTTCGAGTGAGTTG and *B*=AGCTTGAGGTTGAGTGAGTG, *BLAST* is used to get the alignment result shown in the Fig.6a; *Needlman-Wunseh* is used to get one shown in the Fig.6b; *Smith-Waterman* with the result in Fig.6c; and the algorithm in this paper with one in the Fig.6d.

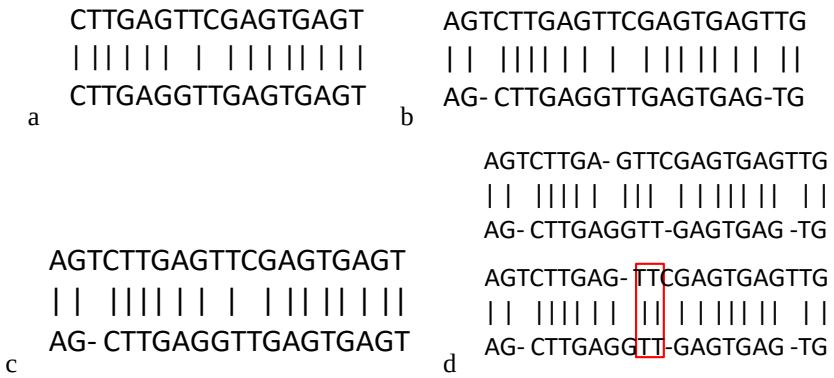


Fig. 6 Results alignment a The alignment result by *BLAST* b The alignment result by *Needle-Wunseh* c The alignment result by Smith-Waterman d The alignment results by the algorithm in the paper

From the above alignment results, there are only 15 pairs of the same bases in the *BLAST* alignment, 17 pairs in *Smith-Water* algorithm, and 18 pairs in *Needle-Wunseh*. When this algorithm is used to deal the alignment, there are 19 pairs same bases. In other words, in the processes of alignment, BLAST concealed parts information, while *Needle-Wunseh* and *Smith-Water* cannot find all the information because of addition or deletion. For example, in Fig.6d two pairs of same bases are marked out by the red color, while in the BLAST algorithm and Smith-Waterman alignment algorithm, they did not match.

In order to compare the merits of the algorithm with all the established ones, seven pairs of *microRNA* sequences with similarity are chosen as the alignment data in *miRBase data*. Firstly, making the global alignment, and then making alignment experiment by the better local alignment results with that in *Needle-Wunseh*. The result is as *Table 1*.

Table 1 the alignment results

Line number	Length	*Needleman-Wunseh* alignment Number	This paper alignment number
hsa-let-7a-1 hsa-let-7a-3	80 74	65	66
hsa-let-7e hsa-let-7g	79 84	61	61
hsa-miR-135a-1 hsa-miR-135b	90 97	69	71
hsa-miR-146a hsa-miR-146b	99 73	53	61
hsa-miR-452 hsa-miR-484	85 79	34	45
hsa-let-7a-2 hsa-let-7a-1	72 80	60	61
hsa-mir-218-1 hsa-mir-218-2	110 110	81	86

From the *table 1*, the proposed algorithm is superior to *Needle-Wunseh* in accuracy since the proposed method reset the conditions of filtering after output all the alignment results, which is most probably able to output the optimal alignment results and to ensure the basic information presence at the same time.

The algorithm is not only applied for those two sequences with high global similarities, but also for the local alignment in most cases. The global alignments may reveal some matching sequence segments which may hide some local similarities, while the algorithm can help compare the local similarities. During the local alignments, tracking should start from the position with high values, which can assist to filter out segments with low similarities, and make it better and faster to find the segments with high similarities.

Just as what the local alignment is shown in the Fig.7a and Fig.7b, if there are two different sequences, *A*=AGTCTTGAGTTCGAGTGAGT and *C*=TGAGTGAGTG, *Smith-Waterman* is used to get the local alignment result shown in Fig.7a, and the algorithm is used to get one in Fig.7b.

```
GAGTGAGT
| | | | | | | |
GAGTGAGT
```

a

```
AGTCTTGAGTTCGAGTGAGTTG
|  | | | | | | |  |
----------T-GAGTGAGT-G
```

b

Fig.7 a The local alignment result with Smith-Waterman b the local alignment result

The proposed algorithm can not only carry out a sequence of global alignment, but also local alignment. In addition, it also solves a problem that an algorithm can only deal with one situation which is the global alignment or the local alignment. Furthermore, it provides the basis for the subsequent analysis of genes, evolution and similarity research of nucleic acid.

4. Discussions

This paper proposed an optimization algorithm based on the backtracking of the binary path matrix which is suitable for both global double sequence alignment and local sequence alignment. A scoring matrix estimated is constructed based on the dynamic programming algorithm and a binary path matrix. Then, the backtracking paths are extracted from the path matrix based on backtracking algorithm, and each backtracking path represents a alignment results. In this algorithm, global and local sequence alignment can be conducted at the same time. Compared with other sequence alignment algorithms, our proposed method can find more bases which are the same ones and provide better basis for analyzing the similarity of sequences.

Acknowledgements

This work is supported by Qiqihar science and technology plan projects(GYGG-201513), and Rese arch on the Teaching Reform of Degree and Postgraduate Education in Heilongjiang in 2016: Resea rch on the Mode of Cultivating Postgraduates in Normal Universities.

References

[1]. Bodenhofer, U., et al., msa: an R package for multiple sequence alignment. Bioinformatics, 2015. 31(24): p. 3997-9.
[2]. Pais, F.S., et al., Assessing the efficiency of multiple sequence alignment programs. Algorithms Mol Biol, 2014. 9(1): p. 4.
[3]. Deorowicz, S., A. Debudaj-Grabysz and A. Gudys, FAMSA: Fast and accurate multiple sequence alignment of huge protein families. Sci Rep, 2016. 6: p. 33964.
[4]. Altschul, S.F., Amino acid substitution matrices from an information theoretic perspective. J Mol Biol, 1991. 219(3): p. 555-65.
[5]. Bashford, D., C. Chothia and A.M. Lesk, Determinants of a protein fold. Unique features of the globin amino acid sequences. J Mol Biol, 1987. 196(1): p. 199-216.
[6]. Ranwez, V., Two Simple and Efficient Algorithms to Compute the SP-Score Objective Function of a Multiple Sequence Alignment. PLoS One, 2016. 11(8): p. e0160043.
[7]. Terrapon, N., et al., Rapid similarity search of proteins using alignments of domain arrangements. Bioinformatics, 2014. 30(2): p. 274-81.
[8]. Needleman, S.B. and C.D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol, 1970. 48(3): p. 443-53.
[9]. Ezawa, K., General continuous-time Markov model of sequence evolution via insertions/deletions: local alignment probability computation. BMC Bioinformatics, 2016. 17(1): p. 397.
[10]. Waterman, M.S., Efficient sequence alignment algorithms. J Theor Biol, 1984. 108(3): p. 333-7.
[11]. Gotoh, O., Alignment of three biological sequences with an efficient traceback procedure. J Theor Biol, 1986. 121(3): p. 327-37.
[12]. Smith, T.F. and M.S. Waterman, Identification of common molecular subsequences. J Mol Biol, 1981. 147(1): p. 195-7.
[13]. Pervez, M.T., et al., Evaluating the accuracy and efficiency of multiple sequence alignment methods. Evol Bioinform Online, 2014. 10: p. 205-17.
[14]. Thompson, J.D., D.G. Higgins and T.J. Gibson, CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Res, 1994. 22(22): p. 4673-80.
[15]. Notredame, C., L. Holm and D.G. Higgins, COFFEE: an objective function for multiple sequence alignments. Bioinformatics, 1998. 14(5): p. 407-22.
[16]. Wang, Y. and K.B. Li, An adaptive and iterative algorithm for refining multiple sequence alignment. Comput Biol Chem, 2004. 28(2): p. 141-8.

[17]. Notredame, C., D.G. Higgins and J. Heringa, T-Coffee: A novel method for fast and accurate multiple sequence alignment. J Mol Biol, 2000. 302(1): p. 205-17.
[18]. Thompson, J.D., et al., Towards a reliable objective function for multiple sequence alignments. J Mol Biol, 2001. 314(4): p. 937-51.
[19]. Notredame, C. and D.G. Higgins, SAGA: sequence alignment by genetic algorithm. Nucleic Acids Res, 1996. 24(8): p. 1515-24.
[20]. Notredame, C., E.A. O'Brien and D.G. Higgins, RAGA: RNA sequence alignment by genetic algorithm. Nucleic Acids Res, 1997. 25(22): p. 4570-80.
[21]. Nguyen, H.D., et al., Aligning multiple protein sequences by parallel hybrid genetic algorithm. Genome Inform, 2002. 13: p. 123-32.
[22]. Roshan, U., Multiple sequence alignment using Probcons and Probalign. Methods Mol Biol, 2014. 1079: p. 147-53.
[23]. Krogh, A., et al., Hidden Markov models in computational biology. Applications to protein modeling. J Mol Biol, 1994. 235(5): p. 1501-31.
[24]. McClure, M.A., C. Smith and P. Elton, Parameterization studies for the SAM and HMMER methods of hidden Markov model generation. Proc Int Conf Intell Syst Mol Biol, 1996. 4: p. 155-64.
[25]. Katoh, K. and D.M. Standley, MAFFT multiple sequence alignment software version 7: improvements in performance and usability. Mol Biol Evol, 2013. 30(4): p. 772-80.
[26]. Liu, Y. and B. Schmidt, Multiple protein sequence alignment with MSAProbs. Methods Mol Biol, 2014. 1079: p. 211-8.
[27]. Havgaard, J.H., et al., Pairwise local structural alignment of RNA sequences with sequence similarity less than 40%. Bioinformatics, 2005. 21(9): p. 1815-24.
[28]. Taneda, A., Multi-objective pairwise RNA sequence alignment. Bioinformatics, 2010. 26(19): p. 2383-90.
[29]. Da, S.F., et al., Parallel Niche Pareto AlineaGA--an evolutionary multiobjective approach on multiple sequence alignment. J Integr Bioinform, 2011. 8(3): p. 174.
[30]. Hung, J.H. and Z. Weng, Sequence Alignment and Homology Search with BLAST and ClustalW. Cold Spring Harb Protoc, 2016.
[31]. Li, J., et al., PSRna: Prediction of small RNA secondary structures based on reverse complementary folding method. J Bioinform Comput Biol, 2016. 14(4): p. 1643001.