Center Star Method for Multiple Sequence Alignment

1. Introduction

Multiple sequence alignment (MSA) plays a very important role in the field of computational biology. Alignment, in bioinformatics, simply means to compare one sequence of nucleotides with others. Sequences that are very much alike, may have the same function, be it a regulatory role in the case of similar DNA molecules; or a similar biochemical function and three dimensional structure in case of the proteins. Moreover, if two sequences of two different organisms are similar then there may exist a relationship of a common ancestor sequence. When this comparison involves only two sequences then it is known as pairwise alignment. When the comparisons consider more than two sequences then it is called multiple sequence alignment.

Multiple sequence alignment has a great value in bioinformatics, because it is central in some important tasks, such as, representation of protein families, identification and representation of conserved features of DNA/protein sequences, detection of homology between a newly sequenced gene and an existing gene family. Moreover, one approach to find the correct family for a newly sequenced unknown protein is to compare the sequence of the protein to the alignment of each family. Another important use of multiple sequence alignment is in the field of phylogenetic study. That is, multiple sequence alignment is used for inferring the evolutionary history.

There are a number of methods for aligning multiple sequences, each of has its own advantages and disadvantages. In this paper, we studied the center star method of multiple sequence alignment. We also implemented it and observed the aligned sequences for large data set.

2. Overview of multiple sequence alignment

Sequence alignment is the procedure of comparing two or more sequences. Two sequences are aligned by putting them into two rows. Identical or similar characters are placed in the same column. The non-identical characters can either be placed in the same column as a mismatch or opposite a gap in the other sequence. In an optimal alignment, non-identical characters and gaps are placed in such a way to make as many identical or similar characters as possible into corresponding columns.

There are two types of sequence alignments: global and local. Global alignment involves the alignment of two entire sequences. Sequences that are quite similar and approximately the same length are suitable candidates for global alignment. In local alignment, portion of sequences with highest density of matches are aligned. Local alignments are more suitable for those sequences of different length that share a conserved region or domain.

Multiple sequence alignment is used in place of pairwise alignment due to the fact that, sometimes pairwise alignment can not discover the hidden relationship between two weakly

similar sequences. By comparing a series of weakly similar sequences, it is possible to observe the relationship between them. The study of multiple alignments is used both for similarity studies and dissimilarity studies. Similarity studies are useful to classify the newly found unknown sequence of gene or protein, where as dissimilarity studies are useful for phylogenetic relationships.

An alignment of two strings X and Y is obtained by first inserting chosen spaces into, or at either end of, X and Y and then placing the two resulting strings one above the other so that every character or space of either string is opposite a unique character or a unique space in the other string. Two opposing identical characters form a match, and two opposing non-identical characters from a mismatch. A space in one string opposite a character x in the second string can also be thought of as a deletion of x from the second string, or an insertion of x into the first string.

Given k sequences $S = \{S_1, S_2, \ldots, S_k\}$. A multiple alignment of S is a set of k equal-length sequences $\{S_1^*, S_2^*, \ldots, S_k^*\}$ where S_i^* is obtained by inserting gaps in to S_i. Such that, each column contains at least one letter from the sequence. That is, no column can contain only dashes (-).

For example:
Consider the following four sequences:

A	T	C	A	
A	T	A	T	A
A	C	C	T	
C	T	T	C	

The multiple sequence alignment of the above set of strings

A	T	C	-	A
A	T	A	T	A
A	C	C	T	-
C	T	-	T	C

3. Approaches of multiple sequence alignment

There is a number of ways to align multiple sequences. Each approach has its own strength and weakness. Following are the well-known approaches of multiple sequence alignments.

1. Dynamic Programming Approach
2. Heuristic Alignment Approach
   - Progressive Approach
     I. Center star method
     II. Tree method
   - Iterative Refinement

Dynamic programming is one of the popular approaches for pairwise alignments. For multiple sequence alignment, the recurrence relation of a dynamic programming for three given sequences and a particular scoring matrix would be

$$S_{i,j,k}=\max\begin{cases} \mathrm{Si}-1,\mathrm{j},\mathrm{k}\ +\ \partial\ (\mathrm{Vi},-,-) \\ Si,j-1,k+\partial\ (-,\mathrm{Wj},-) \\ Si,j,k-1+\partial\ (-,-,\mathrm{Uk}) \\ Si-1,j-1,k+\partial\ (\mathrm{Vi},\mathrm{Wj},-) \\ Si-1,j,k-1+\partial\ (\mathrm{Vi},-,\mathrm{Uk}) \\ Si,j-1,k-1+\partial\ (-,\mathrm{Wj},\mathrm{Uk}) \\ Si-1,j-1,k-1+\partial\ (\mathrm{Vi},\mathrm{Wj},\mathrm{Uk}) \end{cases}$$

Dynamic programming can give the exact solution. Unfortunately, it takes a lot of computation to find the exact alignment. In the case of k sequences, the running time of this approach is $O((2n)^k)$. Dynamic programming is expensive in both time and space. It is rarely used for aligning more than 3 or 4 sequences.

In order to reduce the running time approximation algorithm or heuristics algorithm is used to compute multiple sequence alignments. The basic idea of many of these heuristics is to compute pairwise alignments and to merge alignments consistently. Note that given all pairwise alignments, it is usually impossible to arrange an MSA consistent with all of them. The basic idea of approximation algorithm for aligning multiple sequences is as follows: from a given list of sequences input, S_1, S_2, …, S_n , this approach selects an S_i. Then it builds optimal pairwise alignment between S_i and S_j for each j=1,2,…,n. It uses these pairwise alignments to build a multiple alignment MSA.

4. Center Star Method

For pairwise alignments, we scored each column by looking at matches, mismatches, and gaps in the two sequences. But for multiple sequence alignment, the scoring scheme is a little bit different. There are many possibilities. The sum-of-pairs (SP) is a common scoring scheme. Here, each column in an alignment scored by summing the scores of all pairs of symbols in that column. The score of the entire alignment is then summed over all column scores.

S_1 |	A	C	G	-	-	G	A	G	A
S_2 |	-	C	G	T	T	G	A	C	A
S_3 |	A	C	-	T	-	G	A	-	A
S_4 |	C	C	G	T	T	C	A	C	-

Assume score of match and mismatch/insert/delete are 2 and -2, respectively.

For position 1,
SP-score(A,-,A,C) = δ(A,-) + δ(A,A)+ δ(A,C) + δ(-,A) + δ(-,C) + δ(A,C) = -8
SP-score= -8+12+0+0–6+0+12–10+0 = 0

SP score can be calculated in different ways with different scoring matrix. The performance of the algorithm depends on the choice of calculating SP score. No SP score is suitable for all circumstances.

The idea of the center star alignment is to find a sequence which is most similar to all the rest, and then to use it as the center of a 'star' to align all the other sequences to it.

1. Find $D(S_i, S_j)$ for all i, j.
2. Find the center sequence S_c which minimizes $\sum_{i=1}^{k} D(S_c, S_i)$
3. For every S_i $S_i \in S - \{S_c\}$, choose an optimal alignment between S_c and S_i.
4. Introduce spaces into S_c so that the multiple alignment satisfies the alignments found in previous step.

Consider the five sequences
S_1 CCTGCTGCAG
S_2 GATGTGCCG
S_3 GATGTGCAG
S_4 CCGCTAGCAG
S_5 CCTGTAGG

The first thing according to this algorithm is to choose the centre of the sequences. The following matrix is generated based on the pairwise alignment scores.

	S_1	S_2	S_3	S_4	S_5
S_1	0	4	3	2	4
S_2	4	0	1	6	5
S_3	3	1	0	5	5
S_4	2	6	5	0	4
S_5	4	5	5	4	0

$\sum_{i=1\ldots k} D(S_1, S_i) = 13$
$\sum_{i=1\ldots k} D(S_2, S_i) = 16$
$\sum_{i=1\ldots k} D(S_3, S_i) = 14$
$\sum_{i=1\ldots k} D(S_4, S_i) = 17$
$\sum_{i=1\ldots k} D(S_5, S_i) = 18$

After summing pairwise scores in each row in the matrix, we obtain that S_1 is closest to all the other sequences. Hence, *S*1 is selected to be at the 'center' of the star. Now, this method calculates all the optimal pairwise alignments, the center star sequence, S_1 and all other sequences.

S_1| C C T G C T G C A G
S_2| G A T G - T G C C G

S_1| C C T G C T G C A C
S_3| G A T G - T G C A G

S_1| C C T G C T - G C A G
S_4| C C - G C T A G C A G

S_1| C C T G C T - G C A G
S_5| C C T G - T A G - - G

The next step is to merge all the alignments using "once a gap, always a gap" principle. this process starts with the alignment of S_1 and S_2. Then, adds S_3 *with the previously alligned sequences*. Now, at any stage if the length of upcoming sequence is different with the previously allighned sequences then this method takes the new version of center sequence, S_1 and makes the allignment with all other previously alligned sequences. And this process continues till the last sequence. Thus the final alignment will be:

S_1| C C T G C T - G C A G
S_2| G A T G - T - G C C G
S_3| G A T G - T - G C A G
S_4| C C - G C T A G C A G
S_5| C C T G - T A G - - G

5. Result and Discussion

We implemented the center star method using C++. We observed the output by varying the number of sequences and the size of input data sets. The running time is $O(k^2n^2)$. For calculating the matrix for this method, we used two different ways to calculate SP score. In first case, we aligned all the input sequences and then made the matrix. But, in later case, we made the scoring matrix without aligning the sequences. The first case requires more time than the second case. This is because a considerable amount of time is required to align the sequences.

Sequences	**Data Set**	**Running Time (sec)**
8	2400 x 8	236.17
16	2400 x 16	848.19

Table 1: Calculating SP score with aligned sequences

Sequences	Data Set	Running Time (sec)
8	2400 x 8	63
16	2400 x 16	113
16	5000 x 16	431.45
16	9500 x 16	1623.22
16	15000 x 16	3531.41
24	15000 x 24	5249.50

Table 2: Calculating SP score without aligned sequences

6. Conclusion

We studied the center star method for multiple sequence alignment. We also implemented this method and observed the running time for large data set.

References

1. Neil C. Jones and Pavel A. Pevzner "An introduction to bioinformatics algorithms". MIT Press, 2004.
2. Wing-Kin Sung, "Algorithms in Bioinformatics: A Practical Introduction", CRC Press (Taylor & Francis Group), 2009.
3. Quan Zou, Xiao Shan, Yi Jiang "A Novel Center Star Multiple Sequence Alignment Algorithm Based on Affine Gap Penalty and K-Band" 2012 International Conference on Medical Physics and Biomedical Engineering (ICMPBE2012)
4. Quan Zou, Maozu Guo, Xiaokai Wang, Taotao, Zhang "An Algorithm for DNA Multiple Sequence Alignment Based on Center Star Method and Keyword Tree" Acta Electronica Sinica., 37 (8) (2009), pp. P1746–P1750.
5. Gusfield, Dan. "Efficient methods for multiple sequence alignment with guaranteed error bounds." Bulletin of mathematical biology 55, no. 1 (1993): 141-154.
6. Yong Sun, Zili Zhang, and Jun Wang, "A Novel Algorithm for DNA Multiple Sequence Alignment Based on the Sliding Window and the Keyword Tree," International Journal of Bioscience, Biochemistry and Bioinformatics vol. 3, no. 3, pp. 271-275, 2013
7. http://www.cs.princeton.edu/~mona/Lecture/msa1.pdf last access April 14, 2014
8. http://www.site.uottawa.ca/~lucia/courses/5126-11/lecturenotes/12-13MultipleAlignment.pdf last access April 14, 2014.
9. http://www.cs.tau.ac.il/~rshamir/algmb/98/scribe/pdf/lec05.pdf last access April 14, 2014.