Progressive Alignment using Shortest Common Supersequence

Ankush Garg, Dr. Deepak Garg
Department of Computer Science and Engineering
Thapar University
Patiala, India
ankushgargcse07111@gmail.com, dgarg@thapar.edu

***Abstract*—Multiple Sequence Alignment is an NP-hard problem. The complexity of finding the optimal alignment is $O(L^N)$ where L is the length of the longest sequence and N is the number of sequences. Hence the optimal solution is nearly impossible for most of the datasets. Progressive alignment solves MSA in very economic complexity but does not provide accurate solutions because there is a tradeoff between accuracy and complexity. Guide tree that guides the alignment of sequences is generated by alignment score in progressive alignment. In this paper, Shortest Common Supersequence (SCS) is utilized to generate the guide tree for progressive alignment and the output alignment results are checked by BAliBASE benchmarks for accuracy. According to SP and TC scores, progressive alignment using the guide tree generated by SCS is better than the guide tree generated by alignment score. Original ClustalW2.1 is modified by SCS, and modified ClustalW2.1 gives better results than the original tool.**

Keywords—BAliBASE; ClustalW2; Multiple sequence alignment; Progressive alignment; SCS

I. INTRODUCTION

Multiple Sequence Alignment (MSA) is one of the most significant techniques for comparison between more than two sequences of DNA or protein strands. MSA gives useful data around the sequences by comparing them, such as which gene is responsible for which functionality and structural characteristics in living beings. Dynamic programming can solve the MSA with very high accuracy, but the time increases rapidly with the number and the length of the sequences. Progressive alignment can solve MSA in very economic time, but with less accuracy. In all alignment techniques there is a tradeoff between accuracy and time complexity. After making some changes in progressive alignment and the guide tree generated by alignment score and replacing with guide tree generated by lowest common subsequence score, better outcomes in some instances have been obtained [1]. Lowest common subsequence (LCS) gives less accurate results when the difference between the lengths of sequences is very high. In this paper, progressive alignment using SCS is proposed. SCS of two strings is the shortest string such that both strings are subsequences of it. Progressive alignment not only gives accurate results with a larger difference in lengths of the sequences but also gives accurate results with similar lengths of sequences after using guide tree generated by SCS. BAliBASE 3.0 is used for comparing the results of aligned output. BAliBASE 3.0 contains both benchmark datasets and functions to compare aligned output according to SP and TC scores.

II. LITERATURE REVIEW

Sequence Alignment is the alignment of two strings or sequences by using gap such that the alignment score is maximized. Alignment score increases with the increasing number of matches and decreases with the increasing gaps and mismatches. Sequence Alignment is categorized in two types. The first type is Global Alignment that can be solved by the Needleman-Wunsch algorithm. In Global Alignment, complete sequences are considered, and Alignment Score is globally maximized. The second type is Local Alignment in which parts of sequences are considered, and those parts are called regions of similarity. Local Alignment is very economical but sometimes finding the regions of similarity is a big problem. Local Alignment is locally maximizing the alignment score. Local Alignment is solved by Smith–Waterman algorithm. When the number of sequences is more than two, then alignment is called MSA. MSA is very useful in comparing the DNA and Protein sequences. MSA is used for finding the evolutionary commonality and mutation in the species. There are so many techniques such as progressive alignment, iterative alignment, consistency based, MAFFT and template based which can be used to solve MSA [2].

A. Progressive alignment

Progressive alignment is used to align two sequences or align sets of sequences and repeat this to get complete MSA. It uses a tree which guides that the sequences or set of sequences aligned at which time. This tree which is called the guide tree, can be generated by the alignment score matrix of all individual sequences. Converting score matrix to guide tree is based on the distance between two sequences or set of sequences. Pair of two sequences with better alignment score is aligned earlier. All sequences are in external nodes of the guide tree. According to score, external nodes become siblings and their parent nodes are the alignments of these external nodes. The guide tree is traversed in bottom up manner, and internal nodes guide the alignment of sequences. By this way, all external nodes are connected to the root node, and the root node is the final alignment of all sequences. This alignment has one problem that the alignment depends on the low level

978-1-4799-3080-7/14/$31.00

internal nodes. If the low level internal nodes are suffering from information loss then final alignment also suffers from that information loss. Also, the guide tree by alignment score is not always optimal for MSA. In this way, gaps never decrease in all process. CLUSTAL-W is a general example of progressive alignment [3].

B. Iterative Techniques

Iterative techniques are used to improve accuracy in MSA. There are many techniques for iterative alignment. In these techniques, the accuracy depends on the effectiveness of iteration and number of iterations. Iterative techniques can solve MSA with or without other techniques. Iterative techniques with progressive alignment are used for more accurate results. In iterative techniques with progressive alignment, the sequences are randomly removed, and remaining sequences are again aligned with progressive alignment. This iteration is repeated until certain criteria of the iteration are fulfilled. There is a simple relation between accuracy and time. Accuracy is directly proportional to time and time is directly proportional to the number of iterations [4].

C. Consistency Based Techniques

The larger number of sequences is a big problem. There is the need of scalability for handling larger numbers of sequences than capacity of existing tools. T-Coffee is Tree based Consistency tool for alignment that uses both global and local alignment methods to enhance the accuracy and gives a simple and flexible solution. T-Coffee uses information of all sequences unlike simple Progressive alignment which uses information about current sequences that are being aligned. T-Coffee uses the Balance Guide Tree (BGT) as a guide tree which is used for parallelism. Consistency based tools use the Optimized Library (OLM) which is used to identify the relationship between sequences. T-Coffee also uses the Multiple Tree Alignment (MTA) process which tries to obtain optimal guide tree for best accuracy [5].

D. MAFFT

MAFFT is a MSA tool which reduces the time complexity because this tool uses the fast Fourier transform. MAFFT finds out regions of similarity quickly and gives results in very less time complexity, and it also finds out the score of alignment very quickly. MAFFT is many times faster than T-Coffee. ClustalW uses some methods that grow time complexity according to large input, but MAFFT does not use any complicated function for alignment. MAFFT uses progressive and iterative correction method. MAFFT accuracy is nearly equal to T-Coffee. MAFFT converts protein and DNA sequences into volume and polarity. Different amino acids are assigned to different vectors that have two parts: one is volume and second is polarity [6].

E. Template Based Techniques

Template based techniques use the information about protein and DNA from a template and aligns according to them. The template may be any structural information or any profile. Template is added to the sequences and then the sequences are aligned. In this technique, sequences are checked for a similar template in the existing database. In some tools, users can select a template manually and can also select a method for alignment of sequence [7].

Hidden Markov Model (HMM) is a model in which present state depend on previous states (previous states are hidden). In first order HMM, only last state is considered for the present state. HMM can be used to identify the genes for genome sequencing. Second ordered HMM is used for initial sequencing. States are visible in simple Markov model. HMM contains hidden states, and probability is used for value of hidden states. In HMM, probability is used for aligning the sequences. Frequency distribution is used for insertion and deletion. Insertion and deletion are the states of the model. HMM works on all possible combinations and finds the accurate alignment. In profile HMM sequence alignment, the sequences are searched in a large library of profiles for large scale sequences comparison. Similar sequences are aligned earlier in progressive alignment. Similarity of sequences is scored according to matching profiles of database [12] [13] [4].

BAliBASE is a benchmark that is used to check and compare the accuracy of tools for MSA. BAliBASE 3.0 is the latest version of BAliBASE and has a more refined test case for checking accuracy of tools. BAliBASE has 6255 sequences for testing the accuracy. BAliBASE has a large number of datasets in the form of TFA format and corresponding XML and MSF format given as reference alignment. MSA tools align the dataset given in TFA format and give output in MSF format. If the output of MSA tools is not in MSF format then output is converted to MSF format by help of other tools. BAliBASE compares the output of MSA tools to corresponding reference alignment and output of BAliBASE have SP (sum of pair) score and TC (total column) score. The ratio of correctly aligns pair of sequences is called SP score. The ratio of correctly align columns are called TC score [8].

Subsequence of a string S is a string generated by deleting the characters of the string S. SCS (SCS) of two strings S1 and S2 is the string of lowest length whose subsequences are S1 and S2. Suppose there are three strings “ASDF”, “DFGH” and “ASGH”. The SCS of the above strings is “ASDFGH”. There are various methods for calculating the SCS such as dynamic programming, Bee colony optimization and deposition and reduction algorithm. SCS is used where insertion is the only way to operate the strings. SCS is an NP-Complete problem [9].

Complexity of Progressive alignment is given in three steps. The first step is generating an alignment score for every pair of two nodes. There are N number of sequences, and N* (N-1) /2 is a unique pair of two sequences. Alignment score is generated by a pairwiseAlign function inside FullPairwiseAlign class. The forwardPass function is called to generate a matrix of the alignment scores for two sequences. The reversePass function is used to align the output for two sequences. The complexity for generating alignment score for two sequences is O(L1*L2) where L1 is the length of the first sequence and L2 is the length of the second sequence. Complexity for the first step is $O(L^2N^2)$ where L is the length of the longest sequence and N is the total number of sequences. The second step is generating guide tree from the alignment scores. Distance between two sequences is generated through

the alignment score. Alignment score is divided by length of smaller sequence and then the output is subtracted from 100 and calculated value is divided by 100. Guide tree is generated by distance matrix formed by the above procedure. Complexity for generating one node of the distance matrix is $O(N^2)$. The complexity for generating the guide tree is $O(N^3)$. After generating the guide tree, Progressive alignment is started. Complexity of one iteration of Progressive alignment is $O(NL+L^2)$. The maximum number of iterations required for completing the alignment is N-1 as internal nodes of the guide tree. Hence the complexity of the third step of Progressive alignment is $O(N^2L+NL^2)$. The total complexity is the sum of the complexities of all three steps. The complexity is $O(N^2L^2+N^3)$ [11].

III. Problem Definition

There are N numbers of sequences of DNA or protein strands. The problem is to align the sequences to maximize the alignment score. The score can be calculated by following equation (1) [10]. If the i^{th} character of the first sequence (Si) is compared with the j^{th} character of the second sequence (Sj), then a corresponding score can be calculated as the following equation.

$$Score\ (S_i, S_j) = \begin{cases} 2, \ if Si = Sj \\ 0, \ if Si \neq Sj \ \& \ Si \neq '-' \ \& \ Sj \neq '-' \\ -1, \ if Si \neq Sj \ \& \ Si = '-' \ or \ Sj = '-' \end{cases} \quad (1)$$

From the above equation, the alignment score can be generated but one more penalty for gap opening should be considered. Whenever new gaps are opened then penalty of two is deducted from the alignment score. The total alignment score should be maximized. The gaps should be placed carefully because extra gaps can affect the accuracy. In Progressive alignment, the alignment score between all sequences is generated for generating triangle matrix. A guide tree is generated by triangle matrix according to the value of the alignment scores. S. Hosni et al. explain a new approach for generating guide tree by the help of the LCS (longest common subsequence) [1]. This approach has the drawback that If there are two sequences of length L1 and L2 where L1 is smaller than L2 and the sum of all the lengths of their LCMs is X and the score is 1 – X/L1. This score depends only on L1, but it should depend on L2 or bigger sequence also. For example, the first string is AGAG and second string AGCC than length of the substring is 2 and min length of the string is 4. If the second string is AGCCCC then result will be same. Hence there is a problem because new progressive alignment does not depend on larger string.

IV. Proposed Solution

In Progressive alignment, the guide tree is generated by the alignment score or LCM [1]. SCS of all pairs of two sequences is calculated for generating a triangular matrix. The value in i^{th} row and j^{th} column is the sum of the lengths of both sequences divided by the length of SCS of i^{th} sequence and j^{th} sequence. If the matrix is divided by diagonal then only one triangle (upper triangle of the matrix) is needed to form a guide tree.

$$M_{ij} = \frac{Li+Lj}{LSij} \quad (1)$$

M is the triangular matrix with Mij cells value corresponds to i^{th} row and j^{th} column. Li, Lj and LSij are the length of i^{th} sequence, length of j^{th} sequence and length of SCS of i^{th} and j^{th} sequences respectively. Suppose S1 = AGTCGT, S2 = TCTGA and S3 = AGCTAC are three sequences to be aligned with progressive alignment using the SCS. First of all, matrix M is generated by the help of SCS algorithm [9]. SCS for S1 and S2 is AGTCTGTA, SCS for S1 and S3 is AGCTACGT and SCS for S2 and S3 is TAGCTGAC. LS12, LS13 and LS23 are 1.375, 1.500 and 1.375 respectively. The guide tree (shown in Fig. 1) is guiding to align S1 and S3 after that S2 joined the alignment.

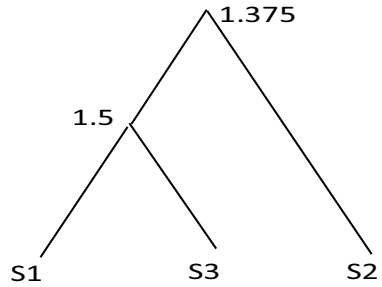

Fig. 1. Guide Tree.

V. Experimental Study

SCS is applied in the original ClustalW2 tool of Progressive alignment. Distance matrix that previously was generated by alignment score is now generated by SCS. In original ClustalW2.1 tool, a guide tree is generated by the distance matrix which is generated through alignment score. In modified ClustalW2.1, a distance matrix is generated by the length of SCS of two strings. The length of SCS is divided by the sum of the length of two corresponding strings. The Full pairwise alignment code is replaced by SCS code. The modified ClustalW2 tool is checked on the basis of BAliBASE 3.0 for accuracy. First 20 sets of RV12 sequences are used for comparing modified ClustalW2 and original ClustalW2.BAliBASE 3.0 is used for comparing reference alignment file in XML format with test alignment in MSF format. The comparison is based on Sum of Pairs (SP) score and the Total Column (TC) score which are represented in Table I. According to Table I, the accuracy of aligned output generated by the modified ClustalW2.1 tool is better than the original ClustalW2.1 tool. The average SP score for all datasets in Table 1 for modified ClustalW2.1 is 0.888 whereas the average SP score for the original ClustalW2.1 tool is 0.878. The average TC score for the modified ClustalW2.1 is 0.769 whereas the average TC score for original ClustalW2.1 is 0.729. Hence the progressive alignment using shortest super sequence is better than traditional Progressive alignment.

In Table I, there is comparison between original ClustalW2.1 (Progressive alignment using the guide tree generated by alignment score) and modified ClustalW2.1 (Progressive alignment using the guide tree generated by SCS).

The comparison is based on SP and TC score using 20 datasets of RV12. On comparing according to SP score, nine entries are showing better results; three entries are showing no change in results and eight entries are showing less accurate results.

TABLE I. SP AND TC SCORE COMPARISON

Sr. No.	Dataset	Original ClustalW2.1		Modified ClustalW2.1	
		SP Score	*TC Score*	*SP Score*	*TC Score*
1	BB12001	0.845	0.650	0.829	0.660
2	BB12002	0.863	0.740	0.898	0.740
3	BB12003	0.865	0.590	0.932	0.890
4	BB12004	0.855	0.550	0.967	0.820
5	BB12005	0.922	0.700	0.914	0.740
6	BB12006	0.959	0.920	0.959	0.920
7	BB12007	0.884	0.700	0.861	0.700
8	BB12008	0.932	0.810	0.910	0.780
9	BB12009	0.919	0.790	0.891	0.750
10	BB12010	0.913	0.810	0.930	0.830
11	BB12011	0.792	0.570	0.777	0.580
12	BB12012	0.708	0.510	0.722	0.530
13	BB12013	0.918	0.810	0.960	0.890
14	BB12014	1.000	1.000	1.000	1.000
15	BB12015	0.550	0.200	0.520	0.200
16	BB12016	0.841	0.680	0.892	0.760
17	BB12017	0.954	0.860	0.959	0.880
18	BB12018	0.958	0.920	0.957	0.920
19	BB12019	0.918	0.850	0.927	0.870
20	BB12020	0.957	0.910	0.957	0.910

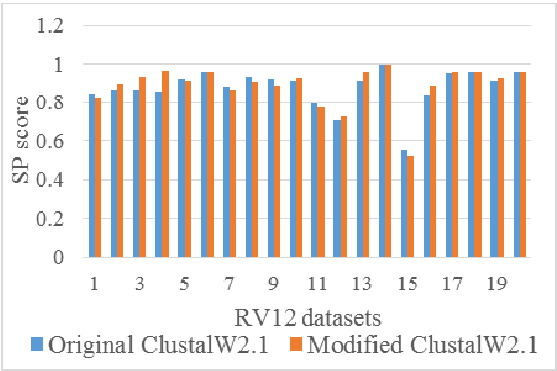

Fig. 2. SP scores comparison by clustered columns chart.

According to TC score, 11 entries are showing better results; seven entries are showing no change and only two entries are showing bad results. Modified ClustalW2.1 is showing acceptable results in 60% of datasets according to SP score and 90% according to TC score. In the fourth entry, there is a remarkable improvement as both SP and TC scores are improved with a big difference. SP score of the original tool for BB12004 is 0.855 and for modified tool, it is 0.967. The difference between both scores is 0.112 which means there are 14% improvements. TC score of the original tool for BB12004 is 0.550 and for modified tool, it is 0.820. The difference between both scores is 0.270 which means 49% improvement. For average cases, there is little improvement in SP and TC scores, but where big difference in the length of sequences, the improvement is remarkable.

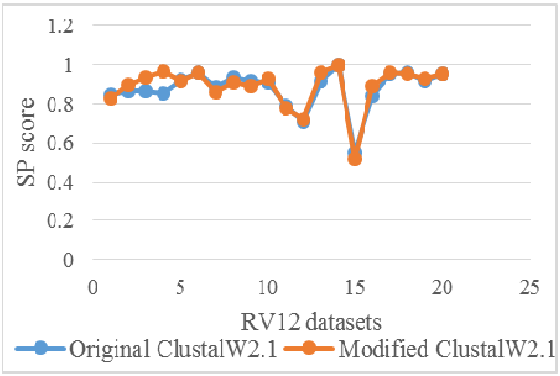

Fig. 3. SP scores comparison by line with markers chart.

In fig. 2 and 3, Bar graph is showing a comparison between the original ClustalW2.1 and the modified ClustalW2.1. Blue bar lines (left side) are showing the SP scores of original ClustalW2.1 and orange bar lines (right side) are showing the SP scores of Modified ClustalW2.1. Sets 2, 3, 4, 16 and 13 are showing remarkable difference. There is improvement in most of the cases, and some cases have excellent improvement. The number of failures is less than the improved cases and decrease in SP score is not large. Modified ClustalW2.1 solves BB12011 with 100% accuracy as compared to the original ClustalW2.1. The decrease in SP score of BB12015 by 0.030 (5.5%) is the worst failure for modified ClustalW2.1. Hence the worst failure is 5.5%, and the best improvement is 14%.

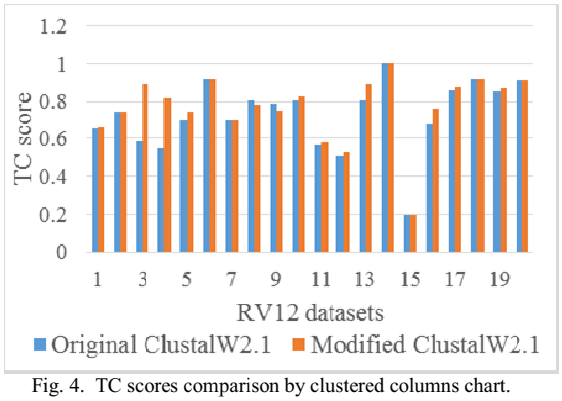

Fig. 4. TC scores comparison by clustered columns chart.

In fig. 4 and 5, the bar graph is showing a comparison between the original ClustalW2.1 and the modified ClustalW2.1. Blue bar lines (left side) are showing the TC scores of the original ClustalW2.1 and orange bar lines (right side) are showing the TC scores of the modified ClustalW2.1. Sets 3, 4, 13 and 16 are showing remarkable difference. An

Output alignment of BB12004 generated by the modified tool is 49% better than the original tool. The worst failure of the modified tool belongs to BB12009 (Decrement of 0.040 in TCscore that is nearly 5%). There are only two failures in TC score, and hence the results of the modified tool are acceptable.

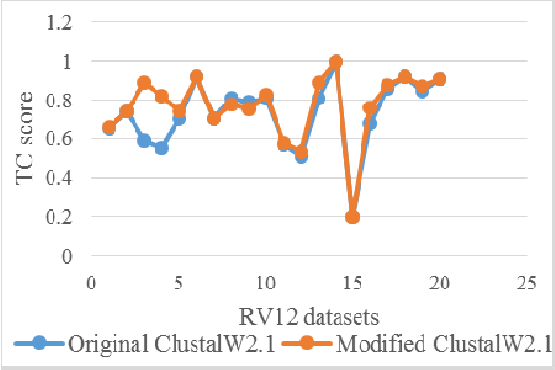

Fig. 5. TC scores comparison by line with markers chart.

The complexity of the modified tool is nearly same as the original tool because complexity for finding SCS is $O(N^2)$ for one pair of sequences, and there are N*(N-1)/2 pairs. Hence complexity for generating distance matrix is $O(N^2L^2)$.

VI. Conclusion and Future Scope

In this paper, changes are done in progressive alignment. The guide tree formed by SCS gives better results in Progressive alignment. SCS indicates that which sequence is more similar and which is less similar. Alignment score does not give better results as the gaps are also considered in normal Progressive alignment. But in Progressive alignment using SCS does not consider gaps in the generation of the guide tree hence gives better results according to similarity. Progressive alignment using SCS considers length of both sequences in pair of two sequences and gives better results. There is a remarkable improvement in TC and SP scores. The complexity of the modified tool is $O(N^2L^2+N^3)$.

There is scope of improvement for special cases. Differences in lengths can measure by the standard deviation and different variants of SCS can be used according to the standard deviation. If the standard deviation is large, then the length of SCS is divided by the sum of the lengths of two corresponding strings otherwise the length of SCS is divided by the length of bigger string.

References

[1] S. Hosni, A. Mokaddem and M. Elloumi, "A new progressive multiple sequence alignment algorithm," *23rdInternational Workshop on Database and Expert Systems Applications (DEXA)*, pp. 195-198, 2012.

[2] R. Chintapalli, A. Kumar and L. Parayitam, "Multiple sequence alignment a quick tour," *15th International Conference on Advanced Computing Technologies (ICACT)*, pp. 1-6, 2013.

[3] S. Isaza, F. Sanchez, G. Gaydadjiev, A. Ramirez and M. Valero, "Scalability analysis of progressive alignment on a multicore," *International Conference on Complex, Intelligent and Software Intensive Systems (CISIS)*, pp. 889-894, 2010.

[4] E. Olson, J. Leonard and S. Teller, "Fast iterative alignment of pose graphs with poor initial estimates," *IEEE International Conference on Robotics and Automation*, pp. 2262-2269, 2006.

[5] M. Orobitg, J. Lladós, F. Guirado, F. Cores and C. Notredame, "Scalability and accuracy improvements of consistency-based multiple sequence alignment tools," In *Proceedings of the 20th European MPI Users' Group Meeting (EuroMPI '13)*. ACM, pp. 259-264, 2013.

[6] K. Katoh, K. Misawa, K. Kuma, and T. Miyata, "MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform,"*Nucleic acids research* 30, pp. 3059-3066, 2002.

[7] A. Fabrice, et al. "Expresso: automatic incorporation of structural information in multiple sequence alignments using 3D-Coffee,"*Nucleic acids research* 34, pp. 604-608, 2006.

[8] J. Thompson, P. Koehl, R. Ripp and O. Poch, "BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark,"*Proteins: Structure, Function, and Bioinformatics* 61(1), pp. 127-136, 2005.

[9] M. Middendorf, "More on the complexity of common superstring and supersequence problems,"*Theoretical Computer Science* 125(2), pp. 205-228, 1994.

[10] S. Lalwani, R. Kumar and N. Gupta, "A study on inertia weight schemes with modified particle swarm optimization algorithm for multiple sequence alignment,"*Sixth International Conference on Contemporary Computing (IC3)* IEEE, pp. 283-288, 2013.

[11] R. Edgar, "MUSCLE: a multiple sequence alignment method with reduced time and space complexity,"*BMC bioinformatics* 5(1), pp. 113-131, 2004.

[12] Srabanti Maji and Deepak Garg, "Hybrid Approach using SVM and MM2 in Splice Site Junction Identification," *Current Bioinformatics Bentham Science* 9(1), pp. 76-85, 2014.

[13] Srabanti Maji and Deepak Garg, "Hidden Markov Model for Splicing Junction Sites Identification in DNA Sequences,"*Current Bioinformatics Bentham Science*, 8(3), pp. 369-379, 2013.

[14] Srabanti Maji and Deepak Garg, "Progress in gene prediction: Principles and challenges,"*Current Bioinformatics Bentham Sciences*, 8(2), 2013.