ISSN: 2321-9653

INTERNATIONAL JOURNAL FOR RESEARCH IN APPLIED SCIENCE AND ENGINEERING TECHNOLOGY (IJRASET)

Analysing Multiple DNA Sequence Alignment Algorithms- Smith Waterman Algorithm and Parallel Smith Waterman Algorithm

Kartika Ahuja[1], Kompal[2]

[#]*Computer Science, DITM Gannaur,DCRUST Murthal*

Abstract— ***Multiple DNA sequence alignment is one of the important research topics of bioinformatics. Rapid and automated sequence analysis facilitates everything from functional classification & structural determination of proteins, to studies of genetic expression and evolution. Rapid and automated sequence analysis facilitates everything from functional classification & structural determination of proteins, to studies of genetic expression and evolution. The ultimate choice of sequence search algorithms is Smith Waterman. However, because of computationally demanding nature of this method, special purpose hardware alternatives of this method like Parallel Smith Waterman have been developed. This keeps the essence of smith waterman with faster computations. In this paper, we present the efficiency of Parallel Smith Waterman over Smith Waterman algorithm***.

Keywords— ***Smith Waterman, Parallel Smith Waterman, DNA sequence Alignment, protein sequence,automated sequence analysis***

I. INTRODUCTION

DNA (Deoxyribonucleic Acid) is the basic unit of an organism, and its length ranges from a few hundred to several billions of nucleotides for different species. Thus, to find out the degrees of similarity and the degrees of difference between DNA sequences is a complicated task. We can see that sequence alignment, especially multiple sequence alignment, is an important research topic of bioinformatics. The results of multiple sequence alignment can be used for other more complicated genetic research. For example, we can use the results of multiple sequence alignment to find out the degrees of similarity of different species, understand the evolutionary history of species, or put a newly found species into a family that it may belong to. In recent years, some methods for multiple sequence alignment have been proposed [1, 3,4, 5, 6, 7, 8].

In [1], Chellapilla et al. presented a method for multiple sequence alignment using evolutionary programming techniques. In [8], Needleman et al. used dynamic programming techniques for multiple sequence alignment.In [3], Isokawa et al. presented a method to deal with multiple sequence alignment using a genetic algorithm (GA). In [4], Notredame et al. presented a method called SAGA for sequence alignment by genetic algorithms. Itinvolved evolving a opulation of alignments in a quasi-evolutionary manner and gradually improving the fitness of the population by an objective function that measures the multiple sequence alignment quality. In [5], Stoye presented techniques of multiple sequence alignment using a divide-and-conquer method, where an increase of the speed compared to optimal multiple alignment by dynamic programming can be guaranteed. In [6], Thompson et al. presented a method for improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positioning specific gap penalties and weight matrix choice.

II. SMITH WATERMAN ALGORITHM

Over a decade after the initial publication of the Needleman-Wunsch algorithm, a modification was made to allow for local alignments (Smith and Waterman, 1981). Today, the Smith-Waterman alignment algorithm is the one used by the Basic Local Alignment Search Tool (BLAST) which is the most cited resource in biomedical literature. In this adaptation, the

ISSN: 2321-9653

alignment path does not need to reach the edges of the search graph, but may begin and end internally. In order to accomplish this, 0 was added as a term in the score calculation described by Needleman and Wunsch.

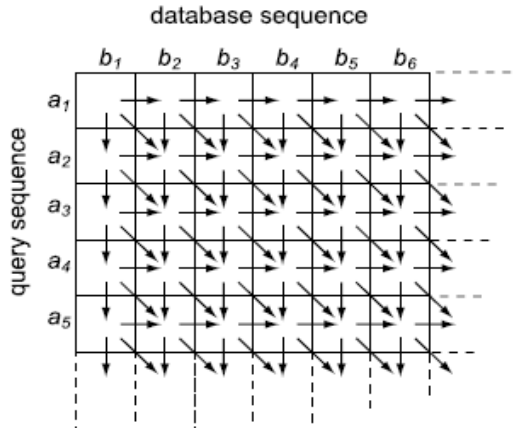

FIG 1- COMPUTATIONAL DEPENDANCIES IN SMITH WATERMAN ALGORITHM

Recall that for global alignments the value at any point is:

$$M(i,j) = MAX(M_{i-1,j-1} + S(A_i, B_j)$$

$$M_{i-1,j} + gap$$

$$M_{i,j-1} + gap)$$

However for local alignments the score calculation becomes:

$$M(I,j) = MAX(M_{i-1,j-1} + S(A_i, B_j)$$

$$M_{i-1,j} + gap$$

$$M_{i,j-1} + gap$$

$$0)$$

The implication of this is that there are no values below zero in a local alignment scoring matrix, and the reason for the zero is exactly the same as in the MSS problem above.

Example:

Find the best local alignment between these two sequences:

ATGCATCCCATGAC

TCTATATCCGT

Using -2 as a gap penalty, -3 as a mismatch penalty, and 2 as the score for a match.

Solution:

Traceback begins at the highest value (which is also the alignment score). Which yields the alignment:

ATCC

||||

ATCC

With an alignment score of 8.

Score = (AA) + (TT) + (CC) + (CC)

= 2 + 2 + 2 + 2

= 8

Local alignments are performed everywhere possible along two sequences.

When trying to find the best local alignments corresponding to a global alignment, a sub-matrix is created with the highest positive score for all alignments above a given threshold. Therefore, the same thing that the MSS was doing on a linear matrix, the Smith-Waterman alignment does on a rectangular matrix.

III. PARALLEL SMITH WATERMAN ALGORITHM

For resolving the critical requirement of memory space, a parallel method for Smith-Waterman algorithm, using the strategy of divide and conquer.

1. Firstly, the original data is partitioned according to the number of processors and distributed to every processor.

ISSN: 2321-9653

2. Secondly the Smith-Waterman algorithm is executed independently on each processor.

3. Thirdly, the intermediate results of each processor are combined and the optimal local alignment is obtained.

Let us see how it works:

1. An unknown sequence usually needs to compare with several known sequences in a sequence database. Often the unknown sequence is called as query sequence and the known sequence is named as subject sequence.

2. Let query sequence be S, subject sequence be T, and the number of processors in a parallel system be p.

3. firstly we divide the query sequence S into p subsequences, described as S0, S1, …, Sp-1, and the length of each subsequence is S/p .

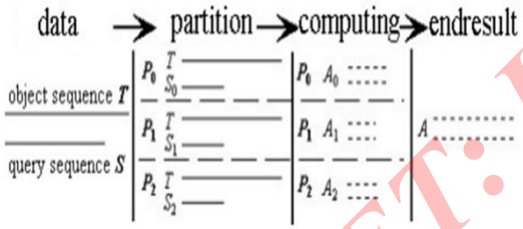

FIG 1 : DIVIDE AND CONQUER METHOD

4. Allocate the query subsequence Si to the processor Pi, where 0<= i<p, and at the same time broadcast the entire subject sequence T to all of the processors .

5. Then, run the Smith-Waterman algorithm for query subsequence Si and subject sequence T in processor Pi independently, and get a local result Ai in processor Pi .

6. Finally, manipulate all of the local result Ai and obtain the final result A of query sequence S and subject sequence T.

This approach uses vectors of cells parallel to the query sequence instead of vectors of cells parallel to the minor diagonal in the matrix.The processor architecture it follows is SIMD ,i.e, Single Instruction Multiple Data stream.This means a single query is sent over multiple data streams by dividing it into parts.

IV. COMPARISON

1. The Smith–Waterman algorithm is fairly demanding of time: To align two sequences of lengths *m* and *n*, *O(mn)* time is required. Smith–Waterman local similarity scores can be calculated in O(m) (linear) space if only the optimal alignment needs to be found, but naive algorithms to produce the alignment require O(mn) space.

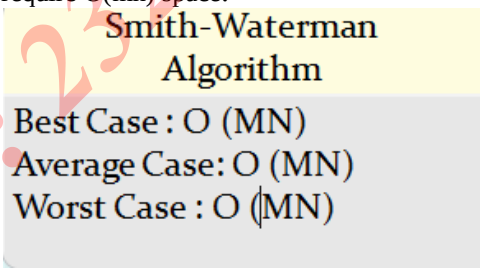

FIG 3 SMITH WATERMAN ALGORITHM'S TIME COMPLEXITY

However, the computational complexities for Parallel Smith Waterman algorithm inversely proportionate to the number of processors p, being used.For two sequences of lengths *m* and *n*, *O(mn/p)* is the run time complexity

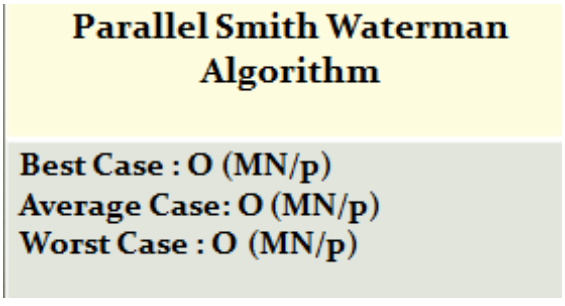

FIG 4 PRALLEL SMITH WATERMAN ALGORITHM'S TIME COMPLEXITY

ISSN: 2321-9653

2. The vector approached used in parallel smith waterman approach is much simplified and faster than traditional smith waterman approach.

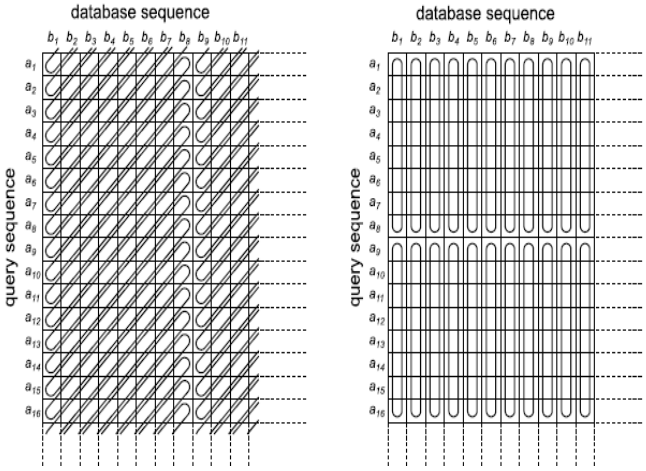

FIGURE 2 : VECTOR ARRANGEMENTS IN TRADITIONAL SMITH WATERMAN AND PARALLEL SMITH WATERMAN

This relates to the faster loading of the vector of substitution scores from memory.

3. However, vector representation puts an overhead of handling data dependencies within the vector. This is the only area where PSW takes a backseat. Therefore for smaller sequences ,PSW should not be used as the overhead of communication between vectors will overcome the computational threshold.
4. The cost of implementing PSW effectively outwits the computational cost of traditional Smith Waterman algorithm.

V. CONCLUSION

Traditional Smith Waterman algorithm and Parallel Smith Waterman algorithm were compared and due to the speed achieved by Parallel Smith Waterman and low cost of implementation, it is by far dubbed as the best DNA sequence alignment algorithm. However this algorithm is limited to long sequence queries and works more effectively on regular region

REFERENCES

[1] Chellapilla, K. and Fogel, G. B. 1999. Multiple sequence alignment using evolutionaryprogramming. Proceedings of the 1999 Congress on Evolutionary *Computation*, Washington D. C.: 445-452.

[2] Torbjorn Rognes.,and Erling Seeburg,Six fold speedup of Smith Waterman algorithm. *Institute of medical microbiology*,University of Oslo NO-0027.

[3] Isokawa, M., Wayama, M., and Shimizu, T. 1996. Multiple sequence alignment using a genetic algorithm. *Proceedings* of the Seventh Workshop on Genome Informatics ,7: 176-177.

[4] Lin, C. H., Chen, S. J., and Chen, S. M 2003. A new method for multiple DNA sequence alignment based on genetic algorithms. Proceedings of the 2003 Joint Conference of AI, Fuzzy System, *and Grey System*, Taipei, Taiwan, Republic of China.

[5] Michalewicz, Z. 1992. "*Genetic Algorithms* + Data Structure = Evolution *Programs*". Springer, Berlin, Germany. sequence of two proteins. *Journal of* Molecular Biology, 48: 443-453.

[6] Setubal, J. and Meidanis, J. 1997. "*Introduction* to Computational Molecular *Biology*". PWS Publishing Company, Boston, U.S.A.

[7] Stoye, J. 1998. Multiple sequence alignment with the divide-and-conquer method. *Gene*, 211: 45-56.

[8] Thompson, J. D., Higgins, D. G., and Gibson, T. J. 1994. CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice. Nucleic Acids Research, 22: 4673-4680.