Journal of Physics: Conference Series

PAPER • OPEN ACCESS

Performance Analysis Of Needleman-Wunsch Algorithm (Global) And Smith-Waterman Algorithm (Local) In Reducing Search Space And Time For Dna Sequence Alignment

Performance Analysis Of Needleman-Wunsch Algorithm (Global) And Smith-Waterman Algorithm (Local) In Reducing Search Space And Time For Dna Sequence Alignment

FN Muhamad[1,a], RB Ahmad[2,b], SM Asi'[1,c] , MN Murad[3,d]

[1]School of Computer & Communication Engineering, Universiti Malaysia Perlis, Perlis.

[2]Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Terengganu

[3]School of Manufacturing Engineering, Universiti Malaysia Perlis, Perlis.

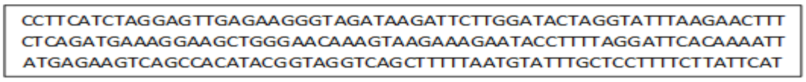
E-Mail: [a]fatimah_noni@yahoo.com, [b]badli@ unisza.edu.my, [c]salina@unimap.edu.my, [d]mnasir@unimap.edu.my

Abstract. Generally, sequence alignment is the process of comparing two sequences to identify similarities and differences between them, then, a typical approach to solve this problem is to find a good and plausible alignment between the two sequences. The data representation in this work is DNA sequence. This work intends to analyze large sequences as well as reducing the search space and time complexity without compromising the accuracy and efficiency. This is by evaluating the performance of Needleman-Wunsch algorithm (global) and Smith-Waterman algorithm (local) based on the Dynamic Programming algorithm. Dynamic Programming algorithm is guaranteed to find optimal alignment by exploring all possible alignments and choosing the best through the scoring and traceback techniques, which is NP-hard to optimize. Implementation of parallel technique using OpenMP on Needleman-Wunsch algorithm and Smith-Waterman algorithm to identify the strengths and weaknesses for both algorithms. By using C Programming, Needle and Smith programs are developed based on the algorithms (respectively). The analysis concluded that the scoring and traceback techniques used in Needle and Smith are able to align an optimal alignment and improved the performance in searching similarity as well as reduced gaps and mismatch. OpenMP directives able to parallelize the codes and execute it faster, with four cores it can get an execution time of around 60% reduced.

Keywords. sequence comparison, sequence alignment, global alignment, local alignment, similarity, optimal alignment, scoring, traceback, match, mismatch, gaps, parallel.

1. Introduction

The data representation in this project is DNA sequence, Deoxyribonucleic Acid is a sequence of string code in which almost all genetic information is encoded, a means of storing biological data. Specifically, the information is encoded using four key chemicals, adenine, thymine, guanine and cytosine (abbreviated as A, T, G and C) [1]. An example of a genome sequence is shown in Figure 1. The DNA sequences of hundreds of organisms have been decoded and stored in databases. This biological sequence data can be obtained from variety of public and private databases.



```
CCTTCATCTAGGAGTTGAGAAGGGTAGATAAGATTCTTGGATACTAGGTATTTAAGAACTTT
CTCAGATGAAAGGAAGCTGGGAACAAAGTAAGAAAGAATACCTTTTAGGATTCACAAAATT
ATGAGAAGTCAGCCACATACGGTAGGTCAGCTTTTTAATGTATTTGCTCCTTTTCTTATTCAT
```

Figure 1. Sub-sequences of DNA for Western Gorilla in FASTA format.

With the growing amount of data, it became impractical to analyze DNA sequences manually, so faster algorithms and tools are needed. Basically, the fundamental procedure of analyzing sequence content is sequence comparison. Sequence comparison is regarded as one of the most fundamental problems of computational biology, which is usually solved with a technique known as sequence alignment. Sequence alignment can be defined as the problem of finding which parts of the sequences are similar and which parts are different. So, a measure of how similar they are is also desirable, then, a typical approach to solve this problem is to find a good and plausible alignment between the two sequences. Finding the optimal alignment is a way to determine the similarity.

Basically, a good alignment has minimum gaps and mismatches inserted into the sequences, but at the same time maximize the number of positions in the aligned strings that match. This problem can then be solved by applying the Dynamic Programming Algorithm that is one of the major strategies for designing algorithms in sequence alignment. The problem of finding optimal alignment can be solved efficiently by determine the best scoring alignment and perform the traceback technique will result a good optimal alignment. To compare two sequences, a scoring system is needed in calculating scores for match, mismatch and gaps. The optimization problem in pairwise sequence alignment is finding the highest scoring (global or local) alignment for a pair of sequences which is optimization problems whose decision versions are NP-complete are called NP-hard. Many of the optimization problems that arise in sequence alignment are NP-hard [2].

Therefore, because of the complexity of the algorithm, there is a need for a methodology that could reduce the computation time while delivering accurate results. These algorithms require quadratic time for each comparison of two sequences [3]. There are many algorithms written that use the approach of Dynamic Programming. However, the global alignment was first introduced in Needleman-Wunsch algorithm based on Dynamic Programming for biological sequences comparison [4]. Later, the improvement of Needleman-Wunsch algorithm proposed Smith-Waterman algorithm to find the best local alignment [5]. The Needleman-Wunsch algorithm and Smith-Waterman algorithm are developed to solve the sequence alignment problems that have computational complexities.

Since the nature of Dynamic Programming requires to store and compute an *mxn* matrix, many researchers spending most of the effort to reduce calculation time. Parallel computing has developed rapidly for scalable data in recent years and has appeared in many Bioinformatics applications [6]. This works also focused on the parallel process using OpenMP. The effort is still being studied by researchers in finding the best way to align sequences by maximize the matches while reducing the gaps, mismatches, length sizes of the optimal alignment and execution time for large data.

2. Literature Review

There are many techniques proposed to solve sequence alignment problems such as Dynamic Programming, Heuristic, Linear Programming, Hidden Markov models and T-Coffee. Optimal alignment is affected by scoring parameters (gaps, mismatch and match), as given a specific scoring system, so, Dynamic programming methods ensure the optimal global alignment by exploring all possible alignments and choose the best [4]. An alternative approach developed by Nordin et al. [7] in five phases include query initialization, patterns generating, pattern's scanning, ranking, and optimal local alignment. The model considers sequences with highest exact matching scores are the most similar to the query sequence. However, many stages of the proposed algorithm require a huge amount of spaces to store results in every stage, which is a source of wasting time.

Zhang, et al. [8], Luis de la Torre and Jaime Seguel, [9] in a characterized work compare most of Dynamic Programming and Heuristic Method sequences alignment algorithms including Needleman-Wunsch, Smith-Waterman, FASTA, and BLAST. The comparisons are aimed to highlight computations and space complexity in term of performance parameters for optimal alignment. These parameters include speed, running times, and affine gap penalties. As a result, there is a tradeoff between speed and sensitivity, BLAST and FASTA consider subsequences k-tuples and words to achieve higher speeds, while Needleman-Wunsch, Smith-Waterman produce optimal and accurate results, but scarifying with the speed in alignment. Parallelization using OpenMP are also used to reduce the computation time. B. Deepa and V. Nagaveni.[10] proposed Needleman-Wunsch algorithm and Smith-Waterman algorithm for identifying the enhanced local sequencing alignments to reduce the running time and increases accuracy of the sequence matching within two sequences. There are also many alignment tools can be obtained from the internet in solving sequence alignment problems for searching similarity such as BLAST, FASTA, CLUSTAL W, LALIGN, GeneWise and EMBOSS.

3. Methodology

The aim of this work is to apply the global and local alignment besides implementing the parallel computing to speed up the process of comparing sequences without compromise with incomplete results, like missing some optimal results. All problems are analysed and studied its characteristic as well as to choose the significant parameters that affect the result. The research methodology as depicted in Figure 2 is designed to produce an effective optimal alignment by reducing the gaps, mismatch and length of sequences optimal alignment without sacrificing the result of accuracy, efficient in time and space complexity, and practical for sequence alignments of large genomic regions. This work designs two proposed algorithms highlighted as, Needle (Needleman-Wunsch algorithm) and Smith (Smith-Waterman algorithm) programs. The programs aligning sequence 1, seqA with *lenA* length (subject sequence) against sequence 2, seqB with *lenB* length (query sequence). To implement the algorithms, a size of matrix (*lenA x lenB*) is calculated.

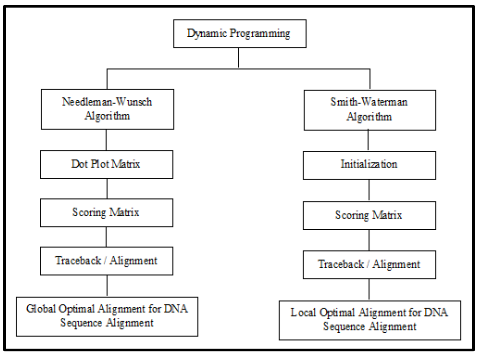


Figure 2. Techniques of Dynamic Programming for Needle program (Parallelization of Needleman-Wunsch algorithm) and Smith program (Parallelization of Smith-Waterman algorithm).

3.1. Needleman_Wunsch Algorithm for Needle Program

All the matrices in parallel version of Needle program for Needleman-Wunsch algorithm are places in global memory space so that all available processors can access them at the same time to perform initialization and other calculations.

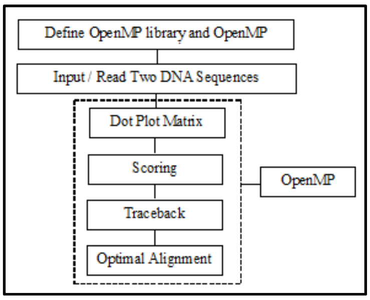

Define OpenMP library and OpenMP
Input / Read Two DNA Sequences
Dot Plot Matrix
Scoring
Traceback
Optimal Alignment
OpenMP

Figure 3. Parallelization of Needle program that run based on the Needleman-Wunsch algorithm.

As illustration in Figure 3, shown the implementation of the parallel programming into the Needle program based on the Needleman-Wunsch algorithm. Needleman-Wunsch can be made to run in linear space, while finding max score is easy but finding path is not.

3.1.2 Dot Plot Matrix of Needle Program. Needle program initialize the *NeedleArray[i][j]* matrix with a Dot Plot matrix. Completed dot plot matrix is using one point (1) for match and zero points (0) for mismatch. Dot plot matrix used parallel pragma in the dot plot process marks the object of *tablefill* and *seq* with parallel shared.

3.1.3 Scoring Matrix of Needle Program. For this work, in cell *NeedleArray[0][0]* is not set to zero, but it is continued with the scoring calculation and all other entries are computed with the scoring formula :

$$NeedleArray[i][j] = \max \begin{cases} NeedleArray[i-1][j-1] + sub(seqA[i], seqB[j]), \\ NeedleArray[i-1][j] + del(seqA[i]), \\ NeedleArray[i][j-1] + ins(seqB[j]) \end{cases}$$

This work it is start in the bottom row moving up, where $i >= 0$ and $j = lenB$ until the process of identifying the cell position is ended. This technique is to simplify in calculating the score by referred to the dot plot matrix. A general rule to calculate the score *NeedleArray[i][j]* is illustrated in Figure 4.

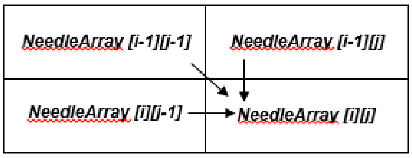


Figure 4. A general rule to calculate the score for Needleman-Wunsch algorithm.

While, parallel processes for scoring are inserting pragma at the outer loop. The scores for *NeedleArray[i][j]* matrix of the alignment of the two sequences by penalty with 0 for a mismatch and a gap and score a match with +1.

3.1.4 Traceback of Needle Program. This work perform a "traceback" by starting at upper-left cell of *NeedleArray[i][j]*, where $i = lenA$ and $j = lenB$, at the highest score. The parallelization of the traceback process is begin with loop that read the scores in the *NeedleArray* matrix until the bottom or far right of the matrix is reached for the size of sequences *lenA* and *lenB*. The traceback process has to identify gap in the alignments, either in Sequence A (*seqA*) or Sequence B (*seqB*).

3.1.5 Optimal Alignment of Needle Program. Needle program produces an optimal sequence alignment based on the Needleman-Wunsch algorithm by exploring all possible alignments and finding the best fit between the two DNA sequences.

3.2. Smith-Waterman Algorithm for Smith Program

There is no constrained to aligning the entire sequences in Smith program. The fact that two zero-length strings is a local alignment with score of 0.

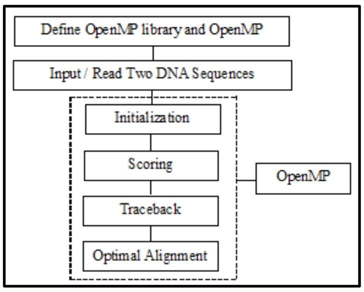

Define OpenMP library and OpenMP
Input / Read Two DNA Sequences
Initialization
Scoring
Traceback
Optimal Alignment
OpenMP

Figure 5. Parallelization of Smith program that run based on the Smith-Waterman algorithm.

As illustration in Figure 5, the implementation of parallel programming to Smith program is based on Smith-Waterman algorithm that uses Dynamic Programming to find the best local alignment between any two given DNA sequences.

3.1.1. Initialization of Smith Program. The first row and column of *SmithArray[i][j]* matrix must be filled with 0. The purpose of the matrix must be filled with 0 is to facilitate the traceback process, where it start with the cell that has the highest score and work back until reached a cell with a score of 0. Besides, the edges of the matrix are initialized to 0 instead of increasing gap penalties. The *SmithArray[i][j]* matrix is first initialized with :

$$SmithArray[i][0] = 0 \text{ and } SmithArray[0][j] = 0, \text{ for all } i \text{ and } j$$

There are parallel double for loop in initialization, so it parallelizes with the outer loop parallel by inserting the private directive into the pragma. Here, can be seen that the execution time did indeed get reduced by half in plotting dot plot in the *SmithArray[i][j]* matrix.

3.1.2. Scoring Matrix of Smith Program. To fill the scores in SmithArray[i][j] matrix, it needs to define the scoring scheme for Match = 1, Mismatch = -1, GapPenalty = 1 and GapExt = 1. The formula for scoring is :

$$SmithArray[i][j] = \max \left\{ \begin{array}{l} 0\,, \\ SmithArray\,[i-1][j-1] \;+\; sub\,(seqA\,[i], seqB\,[j]\,), \\ SmithArray\,[i-1][\,j] \;+\; del\,(seqA\,[i]\,), \\ SmithArray\,[i][\,j-1] \;+\; ins\,(seqB\,[j]\,) \end{array} \right\}$$

A general rule to calculate the score in *SmithArray[i][j]* is illustrated in Figure 6.

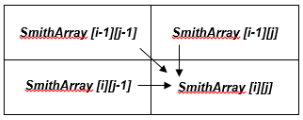


Figure 6. A general rule to calculate the score for Smith-Waterman algorithm.

Parallel insert the pragma at the outer loop, *#pragma* omp parallel shared, and declare shared variables for *tablefill* that shared among the threads. When filling score in the matrix, it do not let any of the matrix values become negative, and thus consider 0 as potentially being the maximum value of the three other cases (where seqA[i] = seqB[j], or there is a gap in seqA or a gap in seqB).

3.1.3. Traceback of Smith Program. To find the optimal alignment it started the traceback with the highest scoring (optimal score) cell in the matrix and it occurs by following the path through the maximum scores back until a 0 value is reached. It finding maximum score in sub row and column of matrix, then trace match in the matrix. When the sequences become aligned their similarity scores align along a diagonal, if there was a gap, the values line up vertically or horizontally.

3.1.4. Optimal Alignment of Smith Program. From the traceback process, it can be seen that when an arrow skips a row, it is a gap in seqA, and when it skips a column, it is a gap in seqB.

3.2. Hardware and Software Development

By using C Programming, Needle and Smith programs are developed based on the algorithms (respectively). For implemented parallel technique on Needle and Smith programs, OpenMP is chosen due to suitability and practically in parallelization. The programs have been tested on HP PC with Intel® Core™ i5-3470 CPU @3.2o GHz for 4 Quad-Core, Memory with 3.2 GiB and LinuxMint 17.1 (rebecca) as the Operating System. The data test is compiled using gcc and run on LinuxMint 17.1 compiler supported from MATE Desktop Environment 1.8.1.

4. Result

Experiments for Needle program (Needleman-Wunsch algorithm) and Smith program (Smith-Waterman algorithm) are conducted on 1000 different pairs of sequences from the Genbank sequence database, National Center for Biotechnology Information (NCBI) (NCBI, 2015). The available data was encoded in the FASTA format (K.D Aumann and R. Huehne, 2008). Dataset in this experiment are made up of DNA sequences from Gorilla gorilla (Western Gorilla) as seqA and Pan Troglodytes Verus (West African Chimpanzee) as seqB. During the sequence alignment process only a DNA sequence from Western Gorilla (seqA) is used and it is held constant. While, seqB consist of 1000 different data from chimpanzees that are run paired with a data from gorilla.

After the data set is analysed, the results parameters of measurement are tabulated. The parameters measured their means, standard deviations and correlation. Spreading of results population within DNA dataset can measured the standard deviation. In determining the effectiveness of the accuracy, space and time it is important to reduce the gap, mismatch and execution time.

The main purpose of the experimental is to evaluate the performance of Needleman-Wunsch and Smith-Waterman algorithms by compared with a publicly available implementation for protein and DNA database searchers, EMBOSS (global) and EMBOSS (local). EMBOSS is an establish system for sequence analysis that has been used quite widely. It is compared to verify the accuracy and consistency of the measurements in space and time complexity.

4.1. Analysis on Similarity

Analysis on similarity generated for Needle against EMBOSS Global and Smith against EMBOSS Local as depicted in Figure 7. According to the result, Needle and Smith programs outperform EMBOSS Global and EMBOSS Local in similarity result. From the analysis, Needle and Smith programs show consistency in similarity result. Needle produced the highest mean with 50.52% where mean is the average of all data plots in the series. The performance of Needle in similarity is better than other methods according to the overall trend of all the 20 datasets. But, in order to determine whether Needle and Smith results are more accurate than others are by comparing both gap and mismatch results.

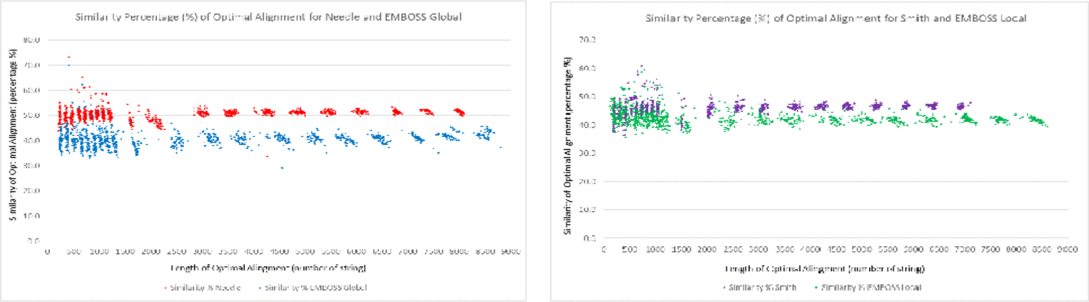


Figure 7. Performance comparison for Similarity (%) between Needle and EMBOSS Global, Smith and EMBOSS Local, respectively.

4.2. Analysis on Gap

The global and local alignment is an alignment such that the matches are maximised and the gaps are minimized [11]. To measure optimal alignment accuracy of Needle and Smith programs, this work compared the output results of gap against EMBOSS Global and EMBOSS Local as shown in Figure 8. According to the result, Needle and Smith programs outperform EMBOSS Global and Local because gap results of Needle and Smith are consistently below from the gap results of EMBOSS Global and Local. The performance of Smith in gap is better than other according to the overall plot trend for the 20 datasets because Smith has the lowest mean of gap, 35.48%.

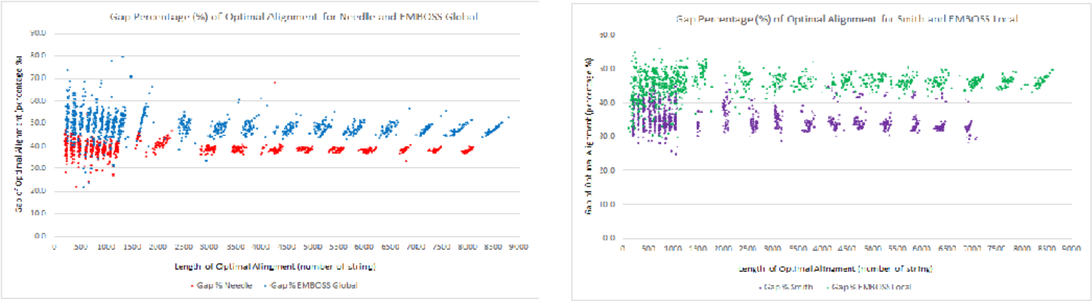


Figure 8. Performance comparison for Gap (%) between Needle and EMBOSS Global, Smith and EMBOSS Local, respectively.

4.3. Analysis on Mismatch

To measure mismatch of Needle and Smith programs, this work compared the output results of mismatch against EMBOSS Global and EMBOSS Local as depicted in Figure 9.

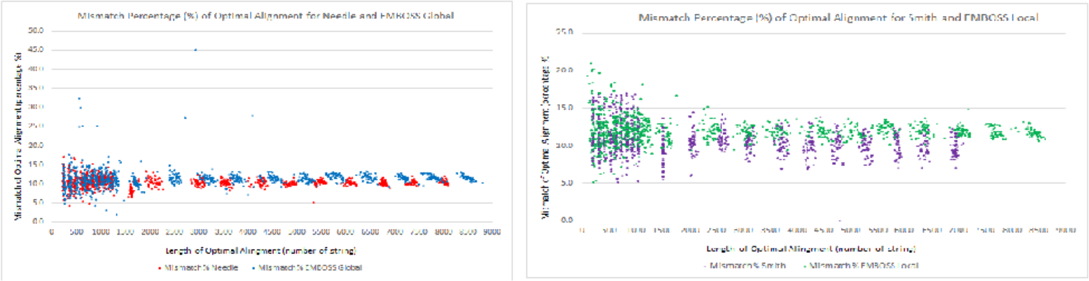


Figure 9. Performance comparison for Mismatch (%) between Needle and EMBOSS Global, Smith and EMBOSS Local, respectively.

According to the result, Needle and Smith programs outperform EMBOSS Global and Local because mismatch result for Needle and Smith are consistently below from the gap result for EMBOSS Global and Local. While Needle has the lowest mean of 10.38%.

4.4. Analysis on Length of Optimal Alignment

To measure length of optimal alignment for Needle and Smith programs, this work compared the output results of optimal length against EMBOSS Global and EMBOSS Local as shown in Figure 10. From the result, Needle and Smith outperform EMBOSS Global and Local in finding the best way to align sequences by searching similarity, reducing gap and mismatch without compromising the sizes of optimal length.

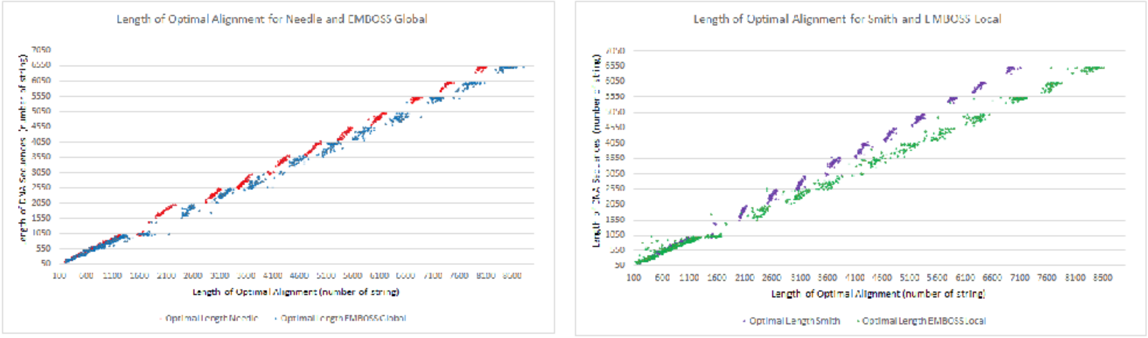


Figure 10. Performance comparison for Lengh of Optimal Alignment between Needle and EMBOSS Global, Smith and EMBOSS Local, respectively.

4.3. Analysis on Execution Time

Figure 11 shows the analysis on execution time is compare the performance of parallel implementation on Needle and Smith using OpenMP. The analysis shows the performance of the larger sequences for Needle and Smith. It is noteworthy to mention that parallelization is very worthwhile for larger sequence, the usage for four and three cores need less time. So, the overhead time is lower than the work that wanted to be parallelized and it is suitable using parallel working under multiple cores.

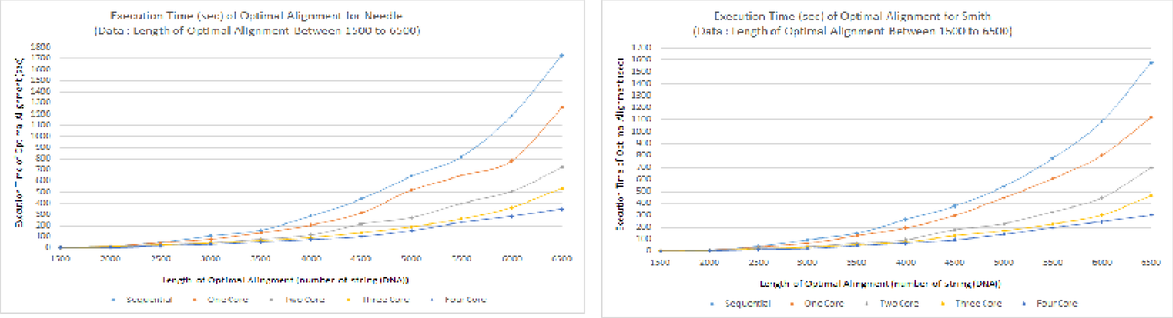


Figure 11. Performance comparison for Execution Time of Optimal Alignment for Needle and Smith from 1500 to 6500 length.

5. Conclusion

With regard to results that gathered from experiment and analysis it shows that Needle and Smith programs are better than EMBOSS Global and EMBOSS Local due to all parameters (gaps, mismatch and match, size of optimal alignment) correlate highly each other in determining accuracy from optimal alignment produced. For larger sequence data, OpenMP directives able to parallelize the code and execute it faster, with four cores it can get an execution time of around 60%. But, for smaller sequences it spent more time to execute the parallel by OpenMP directives.

6. References

[1] S.K. Moore. (2000). Understanding the human genome (vol. 11: pp, 34 – 35). IEEE Press.

[2] Shalini R., Vijay S. and Naveen H. (2011). Bioological computer Model to Solve NP-Complete Problem. International Journal of Information Technology and Knowledge Management. Vol:4, No:1, pp:191-194.

[3] F. Sanchez, E. Salami, A. Ramirez and M.Valero. (2005).Parallel processing in biological sequence comparison usinggeneral purpose processors. Workload Characterization Symposium, 2005. Proceedings of the IEEE International, (pp.99-108).Francis Crick and James Watson. (1953). Molecular Structure Of Nucleic Acids. NATURE, April 25, Volume 171.

[4] S.B. Needleman and Christian D. Wunsch. (1970). A general method applicable to the search for similarities in the amino acid sequence of two sequences. Journal of Molecular Biology, 48(3):443-453.

[5] Temple F. Smith and Michael S. Waterman. (1981). Identification of common molecular subsequences. Journal of Molecular Biology, 147:195-197.

[6] Joshua,L. et al. (2014). BitPAI: A Bit Parallel, General Integer Scoring Sequence Alignment Algorithm. Bioinformatics, 30, 3166–3173

[7] Nordin A., M. Yazid, A. Aziz, and M. Osman. (2009). A guided dynamic programming approach for searching a set of similar DNA sequences. Applications of Digital Information and Web Technologies, 2009. ICADIWT'09. Second International Conference on the, 2009, pp. 512-517.

[8] F. Sanchez, E. Salami, A. Ramirez and M.Valero. (2005).Parallel processing in biological sequence comparison usinggeneral purpose processors. Workload Characterization Symposium, 2005. Proceedings of the IEEE International, (pp.99-108).Francis Crick and James Watson. (1953). Molecular Structure Of Nucleic Acids. NATURE, April 25, Volume 171.

[9] Luis de la Torre and Jaime Seguel, (2015). A Parallel Needleman-Wunsch and Hirschberg Bio-sequence Alignment Algorithm. MIT Press

[10] B. C. Deepa and V. Nagaveni. (2015). Parallel Smith-Waterman Algorithm for Gene Sequencing. International Journal on Recent and Innovation Trends in Computing and Communication. Volume: 3 Issue: 5.

[11] M. Apel, C. Bockermann, and M. Meier. (2009).Measuring similarity of malware behavior. International Proceeding of 34th LCN 2009. IEEE Computer Society.