

VIT[®]

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

REVIEW 3 REPORT

TOPIC: Parallelization of Smith Waterman Local Sequence Alignment Algorithm

Submitted for the course: Parallel and Distributed Computing
(CSE4001)

Submitted by

17BCE2421 RASHI KASERA

17BCE0803 PRANAV MANDALA

17BCE2135 MOHITH J

NAME OF FACULTY: Dr. Sairabanu J

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

CONTENTS:

SL.NO	TOPIC	PAGE.NO
1	ABSTRACT	3
2	KEYWORDS	3
3	INTRODUCTION	4
4	CONTRIBUTION OF THE WORK	5
5	RELATED WORK	6
6	EXISTING SYSTEM DESCRIPTION	11
7	DATA SET DESCRIPTION	13
8	PROPOSED SYSTEM ARCHITECTURE	16
9	CODE	19
10	RESULT	27
11	CONCLUSION	33
12	ACKNOWLEDGMENT	34
13	REFERENCES	35

ABSTRACT:

Genome is an emerging field, constantly presenting many new challenges to researchers in both biological and computational aspect of application. Sequence comparison is a very essential and important operation. They detect similar or identical parts between two sequences called the query sequence and the reference sequence.

The most prevalent kinds of sequence alignment are global and local alignments. In local alignment algorithms we try to match parts of sequences and not the entirety of them. On the other hand, global alignment, we find the superior counterpart between parts of the sequences.

Local alignment is faster than global alignment, due to the lack of need to align the entire sequences. In our project, we would be implementing the Smith-Waterman Algorithm in a serial and parallel manner to for comparison and analysis. As common sense suggests, the parallel implementation should execute and provide the same result as the serial implementation but in a lesser amount of time.

Advances in genomics have triggered a revolution in discovery-based research and systems biology to facilitate understanding of even the most complex biological systems such as the brain.

KEYWORDS:

Local alignment, gene sequencing, smith-waterman algorithm, parallelisation, openMP, SIMD, openMP+SIMD.

INTRODUCTION:

Parallel and Distributed computing is the future of technology. All products and their fundamental concepts are being shifted to a parallel computing model. Everybody would agree that serial computing is easy to implement and use, but simply not efficient enough for industry-level purposes.

Due to this reason, day by day higher number of industries are providing and using cloud solutions which work on the basis on parallel and distributed computing. For instance, Amazon's AWS or Google's Google Cloud platform are becoming the centre for development, may it be in the field of web development, or in the field of data analytics.

To cope up with the fast-paced improvement in technology, one must also become familiarized with this domain, and hence this project. Gene sequencing problem is one of the major issues for researchers regarding optimized system models that could help optimum processing and efficiency without introduction overheads in terms of memory and time.

Bio informatics and computational biology is a latest multidisciplinary field which explains many aspects of the fields of computer science, while computational biology harnesses computational approach and technologies to respond biological questions conveniently.

CONTRIBUTION OF THE WORK:

The work for this project has been planned and distributed equally among our members.

RASHI KASERA (17BCE2421)

The implementation of parallel code was carried out by her. The parallel implementation was done in openMP, openMP+SIMD.

PRANAV MANDALA (17BCE0803)

The implementation of sequential code was carried out by him. The sequential implementation was done in c.

MOHITH J (17BCE2135)

The documentation and theory was carried out by him.

Although the work has been carried out separately, it was well discussed among each other when the work was being done so that everyone has a good knowledge about each others' work.

RELATED WORK:

Sl. no	Title	Author	Journal Name	YEAR	KEY CONCEPTS ADVANTAGES,DISADVANTAGES & FUTURE ENHANCEMENT
1	Sequence Alignment in DNA Using Smith Waterman and Needleman Algorithms	M.P Sudha School of Computing Sciences, Vel's University, Pallavaram, Chennai-600 117 P.Sripriya Assistant Professor, School of Computing Sciences, Vel's University, Pallavaram, Chennai-600 117	International Journal of Computer Science and Information Technologies	2016	This research paper talks about the algorithms for DNA sequencing and scoring patterns. Smith Waterman Algorithm is a local alignment technique which compares nucleic acid sequences. Needleman-Wunsch algorithm is a global alignment technique and finds alignment of the entire sequence of both the proteins. It concludes that most sequences that are homologous share statistical significance.
2	Performance Analysis Of Needleman-Wunsch Algorithm (Global) And Smith-Waterman Algorithm (Local) In Reducing Search Space And Time For Dna Sequence Alignment	FN Muhamad, RB Ahmad, SM Asi, MN Murad	IOP Conf. Series: Journal of Physics: Conf. Series 1019 (2018) 012085	2018	This work intends to analyze large DNA sequences as well as reducing the search space and time complexity without compromising the accuracy and efficiency. 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
3	Analysing Multiple DNA Sequence Alignment Algorithms- Smith Waterman Algorithm and Parallel Smith Waterman	Kartika Ahuja, Kompal	INTERNATIONAL JOURNAL FOR RESEARCH IN APPLIED SCIENCE AND ENGINEERING	2015	This paper states that since DNA Sequence alignment algorithms are computationally demanding in nature special purpose hardware alternatives of this method like Parallel Smith Waterman have been developed. This parallel method for smith waterman algorithm uses strategy of divide and conquer. This keeps

	Algorithm		RING TECHNOL O GY (IJRASET)		the essence of smith waterman with faster computations. However this algorithm is limited to long sequence queries
4	Parallel Smith-Waterman Algorithm for Gene Sequencing	Deepa. B. C , Nagaveni. V	International Journal on Recent and Innovation Trends in Computing and Communication	2015	This paper states the difference between the time complexities of the Smith Waterman algorithm when executed sequentially and in parallel manner. Parallelization of the algorithm reduces the time complexity from $O(mn)$ to $O(m+n)$ for a single sequence comparison and from $O(mnk)$ to $O(m+nk)$ for multiple sequence pairs comparison.
5	Improving the Mapping of Smith-Waterman Sequence Database Searches onto CUDA-Enabled GPUs	Liang-Tsung Huang, Chao-Chin Wu, Lien-Fu Lai and Yun-Ju Li	BioMed Research International Volume 2015, Article ID 185179	June 2015	This paper focuses on how to improve the mapping, especially for short query sequences, by better usage of shared memory. CUDASW++ 3.0 is the latest version, which couples CPU and GPU SIMD instructions and carries out concurrent CPU and GPU computations. Since it is observed that the shared memory in each streaming multiprocessor is not fully utilized in CUDASW++, the execution flow of the Smith-Waterman algorithm was rearranged to fully utilize the shared memory
6	An Improved Smith-Waterman Algorithm Based on Spark Parallelization	Yanfeng Liu Leixiao Li Jing Gao	INT.J.BIO AUTOMATION	March 31, 2019	With the development of big data technology, more and more scholars are now using Spark technology to achieve algorithm parallelization. It was seen that, the rate of the Spark-OSW is positively correlated with the data size within a certain range, i.e., the greater the data, the shorter the Spark-OSW runtime (the greater the SW runtime). For this algorithm, the acceleration in the sixteen-node cluster was much faster than that in the eight-node cluster, revealing that the algorithm performance improves with the growth in the number of nodes.
7	Accelerating Smith-Waterman Alignment for Protein	Yu Liu, Yang Hong, Chun-Yuan Lin, and Che-Lun Hung	International Journal of Genomics	2015 Oct 19	This paper proposes an efficient SW alignment method, called CUDA-SWfr, for the protein database search by using the intratask parallelization technique

	Database Search Using Frequency Distance Filtration Scheme Based on CPU-GPU Collaborative System				based on a CPU-GPU collaborative system. Before doing the SW computations on GPU, a procedure is applied on CPU by using the frequency distance filtration scheme (FDFS) to eliminate the unnecessary alignments. The experimental results indicate that CUDA-SWfr runs 9.6 times and 96 times faster than the CPU-based SW method without and with FDFS, respectively.
8	Comparative Study of the Parallelization of the Smith-Waterman Algorithm on OpenMP and Cuda C	Amadou Chaibou, Oumarou Sie	Journal of Computer and Communications, 2015, 3, 107-117	June 2015	This paper states that the relatively low cost of GP-GPU will make parallel computing more accessible to the public. Expanding the use of GP-GPU to parallel computing in addition to graphics for which they are at the basis created. In the case of the Smith-Waterman algorithm, it is concluded that the GP-GPU accelerates it more than OpenMP. In general, it is recommended to use GP-GPU than OpenMP for massively parallel and long calculations. They have proposed a mathematical modelling of time calculating of the matrix's scoring on the OpenMP and GP-GPU. This equation setting allows us to make wise choices in the number of thread (OpenMP) and the size of the grid computing (GP-GPUs).
9	SWIMM 2.0: Enhanced Smith-Waterman on Intel's Multicore and Manycore Architectures Based on AVX-512 Vector Extensions	Enzo Rucci Carlos Garcia Sanchez	International Journal of Parallel Programming	April 2019	This paper uses SWIMM 2.0 and SIMD exploitation. Most efficient energy footprint implementation.
10	Parallelizing the Smith-Waterman Algorithm using Open	Matthew Baker, Aaron Welch, Manjunath Gorentla Venkata	Oak Ridge National Lab. (ORNL), Oak Ridge,	2015-01-01	Parallelizing the Smith-Waterman algorithm using the OpenSHMEM model and interfaces in OpenSHMEM. The performance characteristics of

	SHMEM and MPI-3 One-Sided Interfaces		TN (United States) Conference Paper		the OpenSHMEM implementation were better than the MPI-3 implementation. The OpenSHMEM implementation using non-blocking interfaces performs better than the implementation using blocking interfaces
11	Parallel Implementation of Smith-Waterman Algorithm using MPI, OpenMP and Hybrid Model	Zeiad El-Saghir, Hamdy Kelash, SayedElnazly, HossamFaheem	International Journal of Innovative Technology and Exploring Engineering (IJITEE)	December 2016	Parallelizing smith-waterman algorithm using different parallel model: pure MPI, pure OpenMP and hybrid MPI/OpenMP model. Hybrid model which combines the MPI and OpenMP models gives better performance in terms of the execution time than the pure MPI model and obviously than the serial time Pure MPI is better than the pure OpenMP model.
12	SWIFOLD: Smith-Waterman implementation on FPGA with OpenCL for long DNA sequences	Enzo Rucci, Carlos Garcia, Guillermo Botella, Armando De Giusti, Marcelo Naiouf & Manuel Prieto-Matias	Selected articles from the 5th International Work-Conference on Bioinformatics and Biomedical Engineering: systems biology	20 November 2018	SWIFOLD: a Smith-Waterman parallel Implementation on FPGA with OpenCL. SWIFOLD offers the best average performance for small and medium test sets, achieving a performance that is independent of input size and sequence similarity. SWIFOLD provides competitive performance rates in comparison with GPU-based implementations on the latest GPU generation for the large dataset.
13	A Case Study for Performance Portability using OpenMP 4.5	Rahulkumar Gayatri Charlene Yang Thorsten Kurth Jack Deslippe	National Energy Research Scientific Computing Center (NERSC) Lawrence Berkeley National Laboratory (LBNL) Berkeley, CA, USA	2015	OpenACC and OpenMP, both lack the ability to fully generate a customized multi-dimensional grid and threads for GPUs. In future, they like to evaluate the practical amount of work required to port kernels which exceed the memory space that can be allocated on the device.
14	An approach of performance comparisons with OpenMP	Chih-Hung Chang Chih-Wei Lu	Wiley Online Library	1 April 2016	They proposed a parallel programming approach using hybrid CUDA and MPI programming, for testing the partition loop iterations according

	and CUDA parallel programming on multicore systems	Chao-Tung Yang Tzu-Chieh Chang			to the number of C1060 GPU nodes in a GPU cluster which consists of one C1060 and one S1070.
15	Parallel Implementation of Smith-Waterman Algorithm using MPI, OpenMP and Hybrid Model	Zeiad El-Saghir, Hamdy Kelash, SayedElnazly, HossamFaheem	International Journal of Innovative Technology and Exploring Engineering(IJITEE)	December 2016	Parallelizing smith-waterman algorithm using different parallel model: pure MPI, pure OpenMP and hybrid MPI/OpenMP model. Hybrid model which combines the MPI and OpenMP models gives better performance in terms of the execution time than the pure MPI model and obviously than the serial time Hybrid model provides better performance than the pure MPI model. Pure MPI is better than the pure OpenMP model.

EXISTING SYSTEM DESCRIPTION:

The Smith–Waterman algorithm performs local sequence alignment; that is, for determining similar regions between two strings of nucleic acid sequences or protein sequences. Instead of looking at the entire sequence, the Smith–Waterman algorithm compares segments of all possible lengths.

Smith–Waterman algorithm aligns two sequences by matches/mismatches (also known as substitutions), insertions, and deletions. Both insertions and deletions are the operations that introduce gaps, which are represented by dashes. The Smith–Waterman algorithm has several steps:

- 1) Initialize the scoring matrix. The dimensions of the scoring matrix are $1 + \text{length of each sequence}$ respectively. All the elements of the first row and the first column are set to 0. The extra first row and first column make it possible to align one sequence to another at any position, and setting them to 0 makes the terminal gap free from penalty.

- 2) Scoring. Score each element from left to right, top to bottom in the matrix. The value of each cell is determined from the cell which has the maximum value among the cell just above it, or to its left or diagonal to it. If the cell which has the maximum value is diagonal to the computing cell we add the value of the diagonal cell and the match value or mismatch value in case of match and mismatch respectively. If the cell which has the maximum value is the cell above it or to its left, we add the gap value to the maximum value. If none of the scores are positive, this element gets a 0. Otherwise the highest score is used and the source of that score is recorded.

- 3) Traceback. Starting at the element with the highest score, traceback based on the source of each score recursively, until 0 is encountered. The segments that have the highest similarity score based on the given scoring system is generated in this process. While performing traceback we write the elements of both gene sequence if the value came from its diagonal element otherwise

we insert a gap denoted by a dash. If the value of that cell came from the cell above it we introduce a gap at that position in the sequence which is written horizontally else if the value of that cell came from the cell to its left we introduce a gap at that position in the sequence which is written vertically. We stop when a 0 is encountered.

Example of a scoring matrix -

		T	G	T	T	A	C	G	G
		0	0	0	0	0	0	0	0
G		0	0	3	1	0	0	0	3
G		0	0	3	1	0	0	0	3
T		0	3	1	6	4	2	0	1
T		0	3	1	4	9	7	5	3
G		0	1	6	4	7	6	4	8
A		0	0	4	3	5	10	8	6
C		0	0	2	1	3	8	13	11
T		0	3	1	5	4	6	11	10
A		0	1	0	3	2	7	9	8

3	6	9	7	10	13
G	T	T	-	A	C
G	T	T	G	A	C

Scoring matrix 1

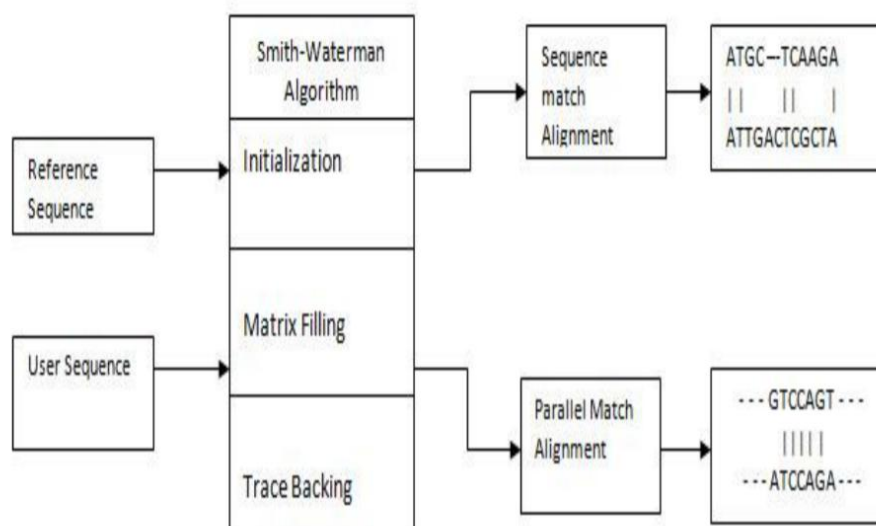


Fig.1 System Architecture

DATA SET DESCRIPTION:

<https://www.ncbi.nlm.nih.gov/nuccore>

10000:

Acytostelium subglobosum LB1 hypothetical protein partial mRNA

NCBI Reference Sequence: XM_012903828.1

[FASTA](#) [Graphics](#)

Go to: ☐

LOCUS XM_012903828 10002 bp mRNA linear INV 15-JUN-2015
DEFINITION Acytostelium subglobosum LB1 hypothetical protein partial mRNA.
ACCESSION XM_012903828
VERSION XM_012903828.1
DBLINK BioProject: [PRJNA280978](#)
BioSample: [SAM00019534](#)
KEYWORDS RefSeq.
SOURCE Acytostelium subglobosum LB1
ORGANISM [Acytostelium subglobosum LB1](#)
Eukaryota; Amoebozoa; Mycetozoa; Dictyosteliida; Acytostelium.

Acytostelium subglobosum LB1 hypothetical protein partial mRNA

NCBI Reference Sequence: XM_012896193.1

[FASTA](#) [Graphics](#)

Go to: ☐

LOCUS XM_012896193 10011 bp mRNA linear INV 15-JUN-2015
DEFINITION Acytostelium subglobosum LB1 hypothetical protein partial mRNA.
ACCESSION XM_012896193
VERSION XM_012896193.1
DBLINK BioProject: [PRJNA280978](#)
BioSample: [SAM00019534](#)
KEYWORDS RefSeq.
SOURCE Acytostelium subglobosum LB1
ORGANISM [Acytostelium subglobosum LB1](#)
Eukaryota; Amoebozoa; Mycetozoa; Dictyosteliida; Acytostelium.

8000:**Dictyostelium discoideum AX4 HEAT repeat-containing protein (DDB_G0279487) mRNA, complete cds**

NCBI Reference Sequence: XM_636563.1

[FASTA](#) [Graphics](#)[Go to:](#) ☐

LOCUS XM_636563 8004 bp mRNA linear INV 29-JAN-2010
DEFINITION Dictyostelium discoideum AX4 HEAT repeat-containing protein
(DDB_G0279487) mRNA, complete cds.
ACCESSION XM_636563
VERSION XM_636563.1
KEYWORDS RefSeq.
SOURCE Dictyostelium discoideum AX4
ORGANISM [Dictyostelium discoideum AX4](#)
Eukaryota; Amoebozoa; Mycetozoa; Dictyosteliida; Dictyostelium.

Acytostelium subglobosum LB1 hypothetical protein partial mRNA

NCBI Reference Sequence: XM_012894172.1

[FASTA](#) [Graphics](#)[Go to:](#) ☐

LOCUS XM_012894172 8022 bp mRNA linear INV 15-JUN-2015
DEFINITION Acytostelium subglobosum LB1 hypothetical protein partial mRNA.
ACCESSION XM_012894172
VERSION XM_012894172.1
DBLINK BioProject: [PRJNA280978](#)
BioSample: [SAM00019534](#)
KEYWORDS RefSeq.
SOURCE Acytostelium subglobosum LB1
ORGANISM [Acytostelium subglobosum LB1](#)
Eukaryota; Amoebozoa; Mycetozoa; Dictyosteliida; Acytostelium.

5000:**Acytostelium subglobosum LB1 hypothetical protein partial mRNA**

NCBI Reference Sequence: XM_012893369.1

[FASTA](#) [Graphics](#)[Go to:](#) 

LOCUS XM_012893369 5004 bp mRNA linear INV 15-JUN-2015
DEFINITION Acytostelium subglobosum LB1 hypothetical protein partial mRNA.
ACCESSION XM_012893369
VERSION XM_012893369.1
DBLINK BioProject: [PRJNA280978](#)
BioSample: [SAMD00019534](#)
KEYWORDS RefSeq.
SOURCE Acytostelium subglobosum LB1
ORGANISM [Acytostelium subglobosum LB1](#)
Eukaryota; Amoebozoa; Mycetozoa; Dictyosteliida; Acytostelium.

Acytostelium subglobosum LB1 hypothetical protein partial mRNA

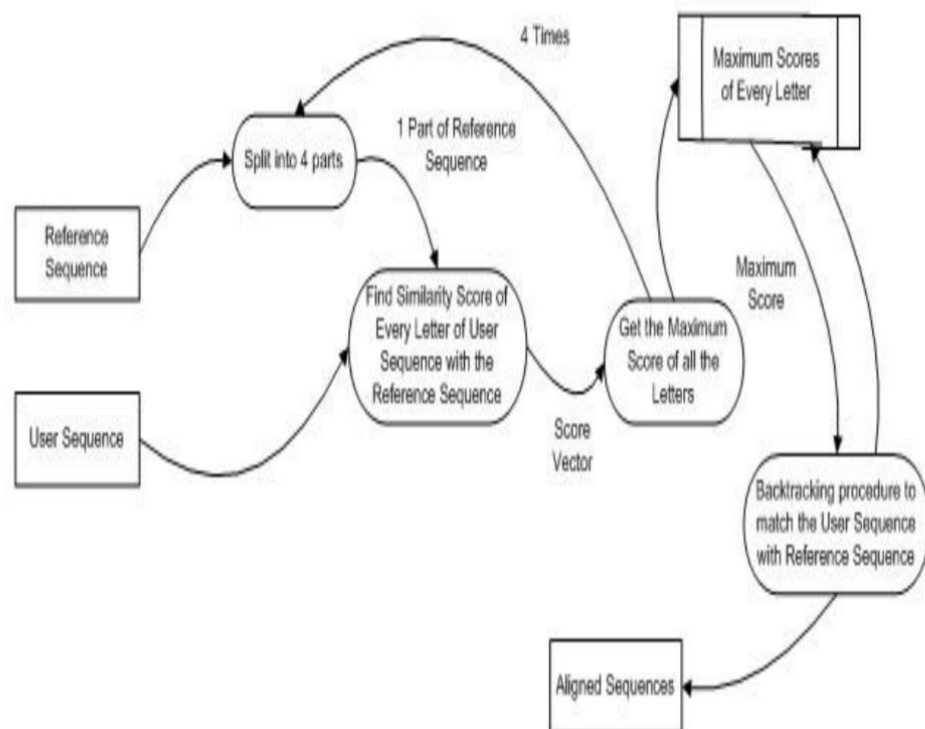
NCBI Reference Sequence: XM_012897382.1

[FASTA](#) [Graphics](#)[Go to:](#) 

LOCUS XM_012897382 5010 bp mRNA linear INV 15-JUN-2015
DEFINITION Acytostelium subglobosum LB1 hypothetical protein partial mRNA.
ACCESSION XM_012897382
VERSION XM_012897382.1
DBLINK BioProject: [PRJNA280978](#)
BioSample: [SAMD00019534](#)
KEYWORDS RefSeq.
SOURCE Acytostelium subglobosum LB1
ORGANISM [Acytostelium subglobosum LB1](#)
Eukaryota; Amoebozoa; Mycetozoa; Dictyosteliida; Acytostelium.

PROPOSED SYSTEM ARCHITECTURE:

In the existing system for local alignment for gene sequencing, sequential process is followed to implement the Smith Waterman algorithm. This process consumes a lot of time when the size of the data is huge. With the advancement in biotechnology it is mandatory that we must be equipped ourselves to process more information as the data being recorded is increasing day to day. Therefore, we are parallelizing the for loops which is used to compute the matrix using openMP and openMP+simd constraints. This helps us reduce the execution time.



Data Flow Diagram 1

So, in this algorithm we can parallelize the part where we are computing the scoring matrix, as we require a nested for loop for computing the scoring matrix.

```
#pragma omp parallel for // (#pragma omp parallel for simd)
```

```
    for(i = 1; i <= lenA; ++i)
```

```
    {
```

```
        #pragma omp parallel for (#pragma omp parallel for simd)
```

```
        for(j = 1; j <= lenB; ++j)
```

```
        {
```

```
            if(FASTA1[i-1] == FASTA2[j-1])
```

```
            {
```

```
                compval = (SWArray[i-1][j-1] + Match);
```

```
            }
```

```
            if(compval < ((SWArray[i-1][j]) + Gap))
```

```
            {
```

```
compval = ((SWArray[i-1][j]) + Gap);
```

```
            }
```

```
            if(compval < (SWArray[i][j-1] + Gap))
```

```
            {
```

```
                compval = ((SWArray[i][j-1]) + Gap);
```

```
            }
```

```
            if(compval < 0)
```

```
            {
```

```
                compval = 0;
```

```
            }
```

```
            if(FASTA1[i-1] != FASTA2[j-1])
```

```
            {
```

```
        if(compval < (SWArray[i-1][j-1] + MissMatch))
        {
            compval = SWArray[i-1][j-1] + MissMatch;
        }
        if(compval < ((SWArray[i-1][j]) + Gap))
        {
            compval = ((SWArray[i-1][j]) + Gap);
        }

        if(compval < ((SWArray[i][j-1]) + Gap))
        {
            compval = ((SWArray[i][j-1]) + Gap);
        }
        if(compval < 0)
        {
            compval = 0;
        }
    }
    SWArray[i][j] = compval;
    compval = 0;
}
}
```

CODE:

CODE (Parallel) for dataset of 5000 :

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <omp.h>
#include <time.h>
FILE *ptr_file_1, *ptr_file_2;

#define TRUE 1
#define FALSE 0
#define Match 3
#define MissMatch -3
#define Gap -2
#define GapExt 0

int inputl;
int StrLen1, StrLen2;
int intcheck = TRUE;

char holder, ch;
int filelen1 = 0;
int filelen2 = 0;
int i,j,k,l,m,n,lenA,lenB,compval;
char dash = '-';

char FASTA1[5000];
char FASTA2[5000];
int HiScore=0;
int HiScorePos[2];
int SWArray[5001][5001];
```

```
char MaxA[5000];
char MaxB[5000];
char OptA[5000];
char OptB[5000];

int MaxAcounter = 1;
int MaxBcounter = 1;
int cont = TRUE;
int check;
int Align(int PosA,int PosB)
{
    int relmax = -1;
    int relmaxpos[2];
    if(SWArray[PosA-1][PosB-1] == 0)
    {
        cont = FALSE;
    }
    while(cont == TRUE)
    {
        for(i=PosA; i>0; --i)
        {
            if(relmax < SWArray[i-1][PosB-1])
            {
                relmax = SWArray[i-1][PosB-1];
                relmaxpos[0]=i-1;
                relmaxpos[1]=PosB-1;
            }
        }
        for(j=PosB; j>0; --j)
        {
            if(relmax < SWArray[PosA-1][j-1])
```

```
{
    relmax = SWArray[PosA-1][j-1];
    relmaxpos[0]=PosA-1;
    relmaxpos[1]=j-1;
}
}
if((relmaxpos[0] == PosA-1) && (relmaxpos[1] == PosB-1))
{
    MaxA[MaxAcounter] = FASTA1[relmaxpos[0]-1];
    ++MaxAcounter;
    MaxB[MaxBcounter] = FASTA2[relmaxpos[1]-1];
    ++MaxBcounter;
}
else
{
    if((relmaxpos[1] == PosB-1) && (relmaxpos[0] != PosA-1))
    {
        for(i=PosA-1; i>relmaxpos[0]-1; --i)
        {
            MaxA[MaxAcounter]= FASTA1[i-1];
            ++MaxAcounter;
        }
        for(j=PosA-1; j>relmaxpos[0]; --j)
        {
            MaxB[MaxBcounter] = dash;
            ++MaxBcounter;
        }
        MaxB[MaxBcounter] = FASTA2[relmaxpos[1]-1];
        ++MaxBcounter;
    }
    if((relmaxpos[0] == PosA-1) && (relmaxpos[1] != PosB-1))
    {
```

```
        for(j=PosB-1; j>relmaxpos[1]-1; --j)
        {
            MaxB[MaxBcounter] = FASTA2[j-1];
            ++MaxBcounter;
        }
        for(i=PosB-1; i>relmaxpos[1]; --i)
        {
            MaxA[MaxAcounter] = dash;
            ++MaxAcounter;
        }
        MaxA[MaxAcounter] = FASTA1[relmaxpos[0]-1];
        ++MaxAcounter;
    }
}
Align(relmaxpos[0],relmaxpos[1]);
}
return(cont);
}
int main()
{
    double s,e;
    s=clock();
    ptr_file_1 = fopen("str2.txt", "r");
    if(ptr_file_1 == NULL)
    {
        printf("Error opening 'str2.txt'\n");
        system("PAUSE");
        exit(1);
    }
    ptr_file_2 = fopen("str1.txt", "r");
    if(ptr_file_2 == NULL)
    {
```

```
printf("Error opening 'str1.txt'\n");
system("PAUSE");
exit(1);
}
fgets(FASTA1, sizeof(FASTA1), ptr_file_1);
fgets(FASTA2, sizeof(FASTA2), ptr_file_2);
fclose(ptr_file_1);
fclose(ptr_file_2);

lenA = strlen(FASTA1);
lenB = strlen(FASTA2);
for(i=0;i<=lenA;++i)
{
    SWArray[0][i]=0;
}
for(i=0;i<=lenB;++i)
{
    SWArray[i][0]=0;
}
compval = 0;
#pragma omp parallel for //pragma omp parallel for simd
for(i = 1; i <= lenA; ++i)
{
    #pragma omp parallel for //pragma omp parallel for simd
    {
        if(FASTA1[i-1] == FASTA2[j-1])
        {
            compval = (SWArray[i-1][j-1] + Match);
        }
        if(compval < ((SWArray[i-1][j]) + Gap))
        {
            compval = ((SWArray[i-1][j]) + Gap);
        }
    }
}
```

```
        if(compval<(SWArray[i][j-1]+Gap))
        {
            compval=((SWArray[i][j-1])+Gap);
        }
        if(compval < 0)
        {
            compval = 0;
        }
        if(FASTA1[i-1] != FASTA2[j-1])
        {
            if(compval < (SWArray[i-1][j-1] + MissMatch))
            {
                compval = SWArray[i-1][j-1] + MissMatch;
            }
            if(compval < ((SWArray[i-1][j]) + Gap))
            {
                compval = ((SWArray[i-1][j]) + Gap);
            }
            if(compval < ((SWArray[i][j-1]) + Gap))
            {
                compval = ((SWArray[i][j-1]) + Gap);
            }
            if(compval < 0)
            {
                compval = 0;
            }
        }
        SWArray[i][j] = compval;
        compval = 0;
    }
}

printf("    0");
```



```
for(i = 0; i <= lenB; ++i)
{
    printf("  %c",FASTA2[i]);
}
printf("\n");
for(i = 0; i <= lenA; ++i)
{
    if(i < 1)
    {
        printf("0");
    }
    if(i > 0)
    {
        printf("%c",FASTA1[i-1]);
    }
    for(j = 0; j <= lenB; ++j)
    {
        printf("%3i",SWArray[i][j]);
    }
    printf("\n");
}
for(i=0; i<=lenA; ++i)
{
    for(j=0; j<=lenB; ++j)
    {
        if(SWArray[i][j] > HiScore)
        {
            HiScore = SWArray[i][j];
            HiScorePos[0]=i;
            HiScorePos[1]=j;
        }
    }
}
```

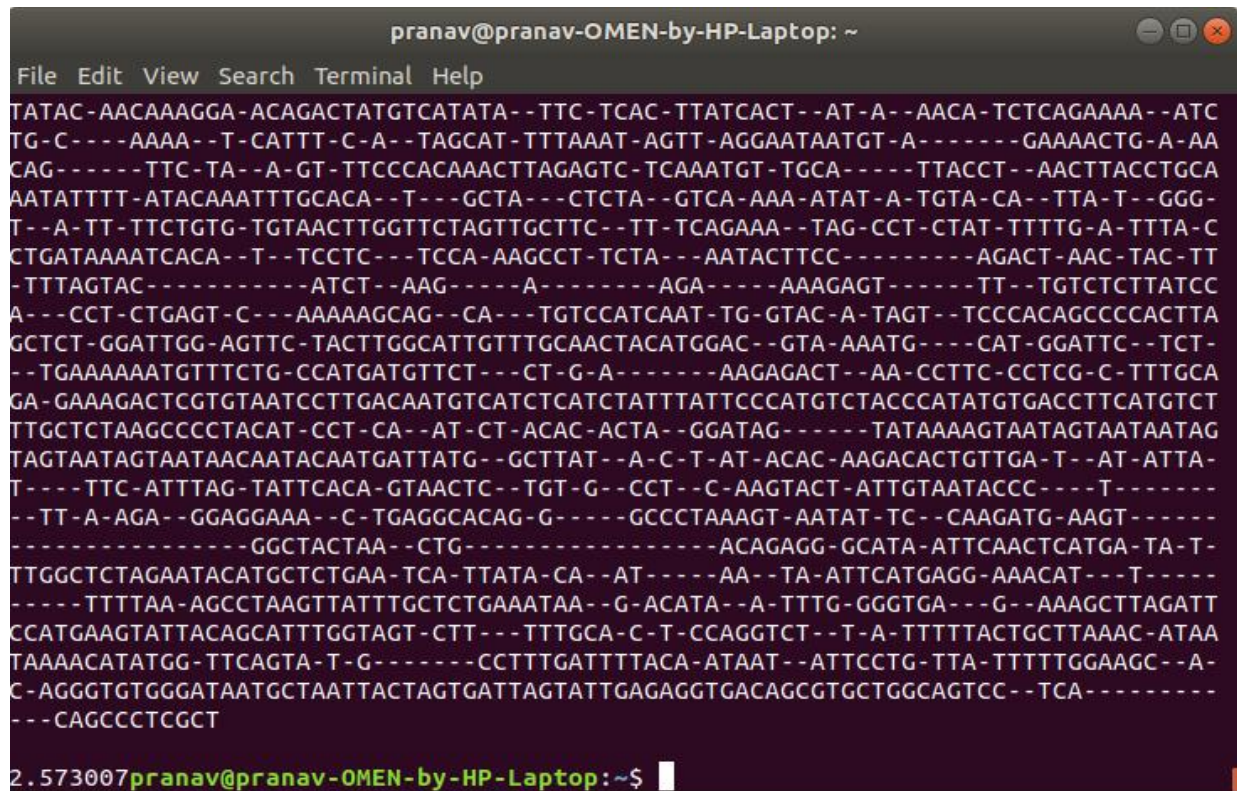
```
    }  
    MaxA[0] = FASTA1[HiScorePos[0]-1];  
    MaxB[0] = FASTA2[HiScorePos[1]-1];  
    check = Align(HiScorePos[0],HiScorePos[1]);  
    k=0;  
    for(i = strlen(MaxA)-1; i > -1; --i)  
    {  
        OptA[k] = MaxA[i];  
        ++k;  
    }  
    k=0;  
    for(j=strlen(MaxB)-1; j > -1; --j)  
    {  
        OptB[k] = MaxB[j];  
        ++k;  
    }  
    printf("%s\n%s ",OptA,OptB);  
    e=clock();  
    printf("\n\n%lf",(double)(e-s)/(double)CLOCKS_PER_SEC);  
    return(0);  
}
```

RESULTS:

SCREENSHOTS:

Data-set: 5000

Serial: 2.669483



```
pranav@pranav-OMEN-by-HP-Laptop: ~  
File Edit View Search Terminal Help  
TATAC-AACAAAGGA-ACAGACTATGTCATATA--TTC-TCAC-TTATCACT--AT-A--AACA-TCTCAGAAAA--ATC  
TG-C----AAAA--T-CATTT-C-A--TAGCAT-TTTAAAT-AGTT-AGGAATAATGT-A-----GAAAACTG-A-AA  
CAG-----TTC-TA--A-GT-TTCCACAAACTTAGAGTC-TCAAATGT-TGCA-----TTACCT--AACTTACCTGCA  
AATATTTT-ATACAAATTTGCACA--T---GCTA--CTCTA--GTCA-AAA-ATAT-A-TGTA-CA--TTA-T--GGG-  
T--A-TT-TTCTGTG-TGTAAGTTGGTTCTAGTTGCTTC--TT-TCAGAAA--TAG-CCT-CTAT-TTTTG-A-TTGA-C  
CTGATAAAATCACA--T--TCCTC--TCCA-AAGCCT-TCTA--AATACTTCC-----AGACT-AAC-TAC-TT  
-TTTAGTAC-----ATCT--AAG-----A-----AGA-----AAAGAGT-----TT--TGTCTCTTATCC  
A--CCT-CTGAGT-C--AAAAAGCAG--CA--TGTCCATCAAT-TG-GTAC-A-TAGT--TCCCACAGCCCCACTTA  
GCTCT-GGATTGG-AGTTC-TACTTGGCATTGTTTGCAACTACATGGAC--GTA-AAATG---CAT-GGATTC--TCT-  
--TGAAAAATGTTTCTG-CCATGATGTTCT--CT-G-A-----AAGAGACT--AA-CCTTC-CCTCG-C-TTTGCA  
GA-GAAAGACTCGTGTAATCCTTGACAATGTCATCTCATCTATTTATTCCCATGTCTACCCATATGTGACCTTCATGTCT  
TTGCTCTAAGCCCCTACAT-CCT-CA--AT-CT-ACAC-ACTA--GGATAG-----TATAAAAGTAATAGTAATAATAG  
TAGTAATAGTAATAACAATACAATGATTATG--GCTTAT--A-C-T-AT-ACAC-AAGACACTGTTGA-T--AT-ATTA-  
T---TTC-ATTTAG-TATTCACA-GTAACTC--TGT-G--CCT--C-AAGTACT-ATTGTAATACCC---T-----  
--TT-A-AGA--GGAGGAAA--C-TGAGGCACAG-G----GCCCTAAAGT-AATAT-TC--CAAGATG-AAGT-----  
-----GGCTACTAA--CTG-----ACAGAGG-GCATA-ATTCAACTCATGA-TA-T-  
TTGGCTCTAGAATACATGCTCTGAA-TCA-TTATA-CA--AT-----AA--TA-ATTCATGAGG-AAACAT---T-----  
-----TTTTAA-AGCCTAAGTTATTTGCTCTGAAATAA--G-ACATA--A-TTTG-GGGTGA---G--AAAGCTTAGATT  
CCATGAAGTATTACAGCATTTGGTAGT-CTT---TTTGCA-C-T-CCAGGTCT--T-A-TTTTACTGCTTAAAC-ATAA  
TAAAACATATGG-TTCAGTA-T-G-----CCTTTGATTTTACA-ATAAT--ATTCCTG-TTA-TTTTGGGAAGC--A-  
C-AGGGTGTGGGATAATGCTAATTACTAGTGATTAGTATTGAGAGGTGACAGCGTGCTGGCAGTCC--TCA-----  
--CAGCCCTCGCT  
2.573007pranav@pranav-OMEN-by-HP-Laptop:~$
```


omp parallel for: 2.635877

```
pranav@pranav-OMEN-by-HP-Laptop: ~
File Edit View Search Terminal Help
TATAC-AACAAAGGA-ACAGACTATGTCATATA--TTC-TCAC-TTATCACT--AT-A--AACA-TCTCAGAAAA--ATC
TG-C---AAAA--T-CATTT-C-A--TAGCAT-TTTAAAT-AGTT-AGGAATAATGT-A-----GAAAACGT-A-AA
CAG-----TTC-TA--A-GT-TTCCACAACTTAGAGTC-TCAAATGT-TGCA-----TTACCT--AACTTACCTGCA
AATATTTT-ATACAAATTTGCACA--T---GCTA--CTCTA--GTCA-AAA-ATAT-A-TGTA-CA--TTA-T---GGG-
T--A-TT-TTCTGTG-TGTAACCTGGTTCTAGTTGCTTC--TT-TCAGAAA--TAG-CCT-CTAT-TTTTG-A-TTTA-C
CTGATAAAATCACA--T--TCCTC--TCCA-AAGCCT-TCTA--AATACTTCC-----AGACT-AAC-TAC-TT
-TTTAGTAC-----ATCT--AAG-----A-----AGA-----AAAGAGT-----TT--TGTCTCTTATCC
A---CCT-CTGAGT-C---AAAAAGCAG--CA---TGTCCATCAAT-TG-GTAC-A-TAGT--TCCCACAGCCCCACTTA
GCTCT-GGATTGG-AGTTC-TACTTGGCATTGTTTGCAACTACATGGAC--GTA-AAATG---CAT-GGATTC--TCT-
--TGAAAAAATGTTTCTG-CCATGATGTTCT--CT-G-A-----AAGAGACT--AA-CCTTC-CCTCG-C-TTTGCA
GA-GAAAGACTCGTGTAATCCTTGACAATGTCATCTCATCTATTATCCCATGTCTACCCATATGTGACCTTCATGTCT
TTGCTCTAAGCCCCTACAT-CCT-CA--AT-CT-ACAC-ACTA--GGATAG-----TATAAAAGTAATAGTAATAATAG
TAGTAATAGTAATAACAATACAATGATTATG--GCTTAT--A-C-T-AT-ACAC-AAGACACTGTTGA-T--AT-ATTA-
T---TTC-ATTTAG-TATTCACA-GTAACTC--TGT-G--CCT--C-AAGTACT-ATTGTAATACCC---T-----
--TT-A-AGA--GGAGGAAA--C-TGAGGCACAG-G-----GCCCTAAAGT-AATAT-TC--CAAGATG-AAGT-----
-----GGCTACTAA--CTG-----ACAGAGG-GCATA-ATTCAACTCATGA-TA-T-
TTGGCTCTAGAATACATGCTCTGAA-TCA-TTATA-CA--AT-----AA--TA-ATTCATGAGG-AAACAT---T-----
-----TTTTAA-AGCCTAAGTTATTTGCTCTGAAATAA--G-ACATA--A-TTTG-GGGTGA---G--AAAGCTTAGATT
CCATGAAGTATTACAGCATTGTTGTTAGT-CTT--TTTGCA-C-T-CCAGGTCT--T-A-TTTTACTGCTTAAAC-ATAA
TAAACATATGG-TTCAGTA-T-G-----CCTTTGATTTTACA-ATAAT--ATTCCTG-TTA-TTTTGGGAAGC--A-
C-AGGGTGTGGGATAATGCTAATTACTAGTGATTAGTATTGAGAGGTGACAGCGTGTGGCAGTCC--TCA-----
---CAGCCCTCGCT
2.213429pranav@pranav-OMEN-by-HP-Laptop:~$
```

simd parallel: 2.487348

```
pranav@pranav-OMEN-by-HP-Laptop: ~
File Edit View Search Terminal Help
TATAC-AACAAAGGA-ACAGACTATGTCATATA--TTC-TCAC-TTATCACT--AT-A--AACA-TCTCAGAAAA--ATC
TG-C---AAAA--T-CATTT-C-A--TAGCAT-TTTAAAT-AGTT-AGGAATAATGT-A-----GAAAACGT-A-AA
CAG-----TTC-TA--A-GT-TTCCACAACTTAGAGTC-TCAAATGT-TGCA-----TTACCT--AACTTACCTGCA
AATATTTT-ATACAAATTTGCACA--T---GCTA--CTCTA--GTCA-AAA-ATAT-A-TGTA-CA--TTA-T---GGG-
T--A-TT-TTCTGTG-TGTAACCTGGTTCTAGTTGCTTC--TT-TCAGAAA--TAG-CCT-CTAT-TTTTG-A-TTTA-C
CTGATAAAATCACA--T--TCCTC--TCCA-AAGCCT-TCTA--AATACTTCC-----AGACT-AAC-TAC-TT
-TTTAGTAC-----ATCT--AAG-----A-----AGA-----AAAGAGT-----TT--TGTCTCTTATCC
A---CCT-CTGAGT-C---AAAAAGCAG--CA---TGTCCATCAAT-TG-GTAC-A-TAGT--TCCCACAGCCCCACTTA
GCTCT-GGATTGG-AGTTC-TACTTGGCATTGTTTGCAACTACATGGAC--GTA-AAATG---CAT-GGATTC--TCT-
--TGAAAAAATGTTTCTG-CCATGATGTTCT--CT-G-A-----AAGAGACT--AA-CCTTC-CCTCG-C-TTTGCA
GA-GAAAGACTCGTGTAATCCTTGACAATGTCATCTCATCTATTATCCCATGTCTACCCATATGTGACCTTCATGTCT
TTGCTCTAAGCCCCTACAT-CCT-CA--AT-CT-ACAC-ACTA--GGATAG-----TATAAAAGTAATAGTAATAATAG
TAGTAATAGTAATAACAATACAATGATTATG--GCTTAT--A-C-T-AT-ACAC-AAGACACTGTTGA-T--AT-ATTA-
T---TTC-ATTTAG-TATTCACA-GTAACTC--TGT-G--CCT--C-AAGTACT-ATTGTAATACCC---T-----
--TT-A-AGA--GGAGGAAA--C-TGAGGCACAG-G-----GCCCTAAAGT-AATAT-TC--CAAGATG-AAGT-----
-----GGCTACTAA--CTG-----ACAGAGG-GCATA-ATTCAACTCATGA-TA-T-
TTGGCTCTAGAATACATGCTCTGAA-TCA-TTATA-CA--AT-----AA--TA-ATTCATGAGG-AAACAT---T-----
-----TTTTAA-AGCCTAAGTTATTTGCTCTGAAATAA--G-ACATA--A-TTTG-GGGTGA---G--AAAGCTTAGATT
CCATGAAGTATTACAGCATTGTTGTTAGT-CTT--TTTGCA-C-T-CCAGGTCT--T-A-TTTTACTGCTTAAAC-ATAA
TAAACATATGG-TTCAGTA-T-G-----CCTTTGATTTTACA-ATAAT--ATTCCTG-TTA-TTTTGGGAAGC--A-
C-AGGGTGTGGGATAATGCTAATTACTAGTGATTAGTATTGAGAGGTGACAGCGTGTGGCAGTCC--TCA-----
---CAGCCCTCGCT
2.159456pranav@pranav-OMEN-by-HP-Laptop:~$
```


simd parallel: 6.236649

```
pranav@pranav-OMEN-by-HP-Laptop: ~
File Edit View Search Terminal Help
ATTAATGGCACAAGTCATGTTTTGATGTTGCACATTTGTCTTTATTTGTGGCTTGTTTTGCTTCCACATCAATCCACTC
AAGGCCTACATTCTGCTATAATGCAATTTCAAGTTCTTTACAGGCCGAGAAAAATGAATCTGAATTCCTGACCTCCAAAA
GTGATCAAGATATTTTTAGTTCAAGCTCCAAAATTTTCTCATTTTCATAGGTTTTCTCGATTGATCATTATTCATGATT
TGCAAGGAATCATTCAATGTTTTCTAAATCTATTACTGCATCCTGACACATATGACATTTTAACTATGTTCCAGATTTTT
GAATGAAGAGTGTAATTTTAAATGTTTTCAACCACAAAAAATAAGTATGTGAAGTGGTGGATTGTTAATTAGCCTTATT
TAACCATTTAATATTGTACACGTACACCAAGCATCATGTTGTACCCCATGAATAC--A-CACAATTATTATTTGTC--
AATTTAAA--AT--GAAATAATAAAAAATAACAAAGGCATTAGCCTCTGCATTGCCTTTACCGGTCATCCTCACGGTGAC
TAACGCAAAAAACGTTCTATTTTCATCCTTACAAACATCCCTATCTTTGATGCCTCTTTGTCTAGATCTCTATCCCCTCCT
GTTTTCTCTACGTTATTTATATGGGTATCA--TCACCATCC--TGGACAA-CATCAGGAC-A-----GATA
TC-CCTC--A--CCA-AGCCAATG-TTCCTCTCT-ATG-TTG--GCTCAA-A-TGTC-C-TTG-AACTTTC-----CTTT
CACCAC--CCTT-T--CCACAG-----TCAAAAG-----GATATTGTAGTTTAAATGCCTCAGAGTTCAGCTT--
-----T-TAAGCTTCTGA-CAAATTATTCTT--CCTCTT--TAGGTT----CTC----CTTT-A-TGG-AATCTT----
-----CTGT-A-----C-----TG-ATGG-CCA--TGTCTTTA-----ACTACT-ATGTAGA-TATC--
T-GCT-ACTA-CCTGTATTAT-G--CCTCTACC-TT--TATTAGCAG--AGTTA-TCTGT-ACT-GTTGGC-ATGAC-AA
TCA---TTTG-----T-TAATATGAC-TTGCC-T-----TTCCTTTTCTGC-----TAT-T-CTTGATCAAA
TGGC--T--CCTC-----TTTCTTGCTCCTCTCATTTCTCCTGCCTTCACTTGGACGTGCTTCACGTAGTCTG-TGCTT
ATG-ACTGGATTAATAA--TTGA--T--ATGGACTTAT-C--CTAA-T-GTTGTTTCGTCATAATATGGGT-----
-----TTTA-----T-GGT-CCATTATTATTTCTATGCATTGATCTGGAG-A-AGGCT--
-T-C-AATCC--TTTT-A-CTCTTTGTGGAATAA-----TC--TGTAACCTTCTG-GT--TCACTC--TGC-TAT
AGC-AATT-----T-CAGTTTAGGCTAGT-AAGCATG-AGGATGCCTCCTTCTCTGATTTTCCCACAGTCTGTTGGT
CACAGAATAACCTGAGTGA-TTACTGATG--AAAG--AGTGAGAA-TGT-T-ATTGA-TAGTC-AC-AATGACA-AAAA
CA-A-ACAAC--ACAG-TCA-----AAA-T
5.211665pranav@pranav-OMEN-by-HP-Laptop:~$
```

Data-set: 10000

Serial: 10.060523

```
pranav@pranav-OMEN-by-HP-Laptop: ~
File Edit View Search Terminal Help
GCCTTTAAGAAGTGAACACTACACGAGGGTCCCGGCTTCATTCTTGAAGTCAGTGAAACCAAGAACCACCAATTCCGGAC
ACAGTATGTCAGAAACAATATGAGTCACTAAATCAATATACTTCTCAACAATTTCCAACAGCCCTTGCAATTAACCTGGCCATGT
GACTGGTTGTGACTAAAATAATGTGGAGATAATAATGTGTACTCCCTAAGGCAGAGTGCCCTTCTATCATTCTCTTTCCCTTCC
TCTATGTGGCAGAAAGTAAAGATTCTGAAATGATAAAGTCAATCACAGGAAGGCACCTGGACTCCTGGCCCACTGCTTGGAGGA
GAGCACTCAGGACCATGAACATCTGACTGTGACGTAGCAATAAAGAAACCCACGTTTCATATGAAACTGCTTAAATTAATGGCA
CAAGTCATGTTTTGATGTTGCACATTTGTCTTTATTTGTGGCTTGTTTTGCTTCCACATCAATCCACTCAAGGCCTACATTCTG
CTATAATGCAATTTCAAGTTCTTTACAGGCCGAGAAAAATGAATCTGAATTCCTGACCTCCAAAAGTGATCAAGATATTTTTAGT
TCAGGCTCCAAAATTTCTCATTTTCATAGGTTTTCTCGATTGATCATTATTCATGATTGCAAGGAATCATTCAATGTTTTCT
AAATCTATTACTGCATCCTGACACATATGACATTTTAACTATGTTCCAGATTTTGAATGAAGAGTGTAATTTTAAATGTTTTT
ACCACAAAAAATAAGTATGTGAAGTGGTGGATTGTTAATTAGCCTTATTTAACCATTTAATATTGTACACGTACACCAAGCAT
CATGTTGTACCCCATGAATAC--A-CACAATTATTATTTGTC--AATTTAAA--AT--GAAATAATAAAAAATAACAAAGGCAT
TAGCCTCTGCATTGCCTTTACCGGTCATCCTCACGGTGACTAACGCAAAAAACGTTCTATTTTCATCCTTACAAACATCCCTATC
TTGATGCCTCTTTGTCTAGATCTCTATCCCCTCCTGTTTTCTCTACGTTATTTATATGGGTATCA--TCACCATCC--TGGACAA
-CATCAGGAC-A-----GATATC-CCTC--A--CCA-AGCCAATG-TTCCTCTCT-ATG-TTG--GCTCAA-A-TG
TC-C-TTG-AACTTTC-----CTTTCACCAC--CCTT-T--CCACAG-----TCAAAAG-----GATATTGTAGTTTAA
TGCCTCAGAGTTCAGCTT-----T-TAAGCTTCTGA-CAAATTATTCTT--CCTCTT--TAGGTT----CTC----CTTT-A
-TGG-AATCTT-----CTGT-A-----C-----TG-ATGG-CCA--TGTCTTTA-----ACTACT-ATGT
AGA-TATC--T-GCT-ACTA-CCTGTATTAT-G--CCTCTACC-TT--TATTAGCAG--AGTTA-TCTGT-ACT-GTTGGC-ATG
AC-AATCA---TTTG-----T-TAATATGAC-TTGCC-T-----TTCCTTTTCTGC-----TAT-T-CTTGATCAAA
TGGC--T--CCTC-----TTTCTTGCTCCTCTCATTTCTCCTGCCTTCACTTGGACGTGCTTCACGTAGTCTG-TGCTTATG-A
CTGGATTAATAA--TTGA--T--ATGGACTTAT-C--CTAA-T-GTTGTTTCGTCATAATATGGGT-----
-----TTTA-----T-GGT-CCATTATTATTTCTATGCATTGATCTGGAG-A-AGGCT--T-C-AATCC--TTT
T-A-CTCTTTGTGGAATAA-----TC--TGTAACCTTCTG-GT--TCACTC--TGC-TATAGC-AATT-----T-CAG
TTTAGGCTAGT-AAGCATG-AGGATGCCTCCTCTCTGATTTTCCCACAGTCTGTTGGTCACAGAATAACCTGAGTGA-TTACT
GATG--AAAG--AGTGAGAA-TGT-T-ATTGA-TAGTC-AC-AATGACA-AAAAACA-A-ACAAC--ACAG-TCA-----
-----AAA-T
8.737021pranav@pranav-OMEN-by-HP-Laptop:~$
```


omp parallel for: 9.902398

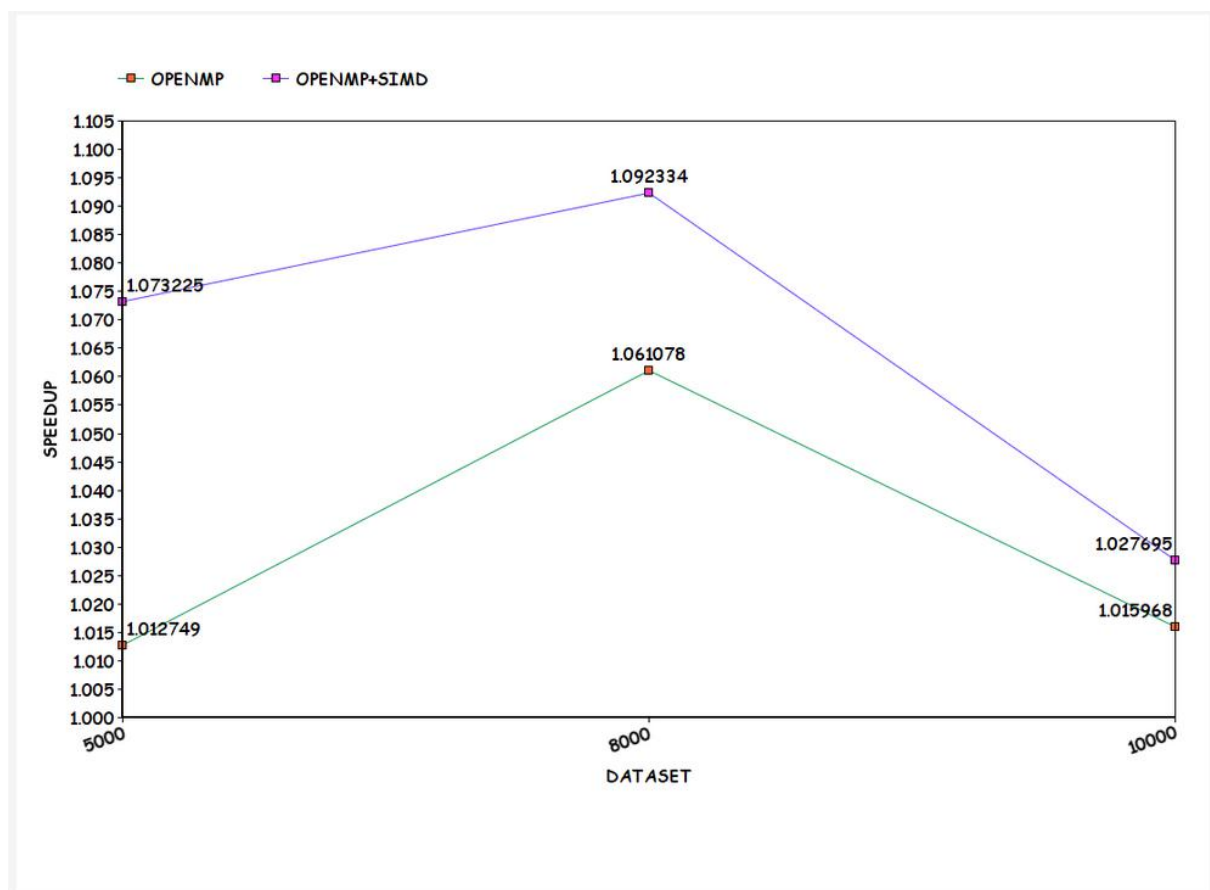
```
pranav@pranav-OMEN-by-HP-Laptop: ~
File Edit View Search Terminal Help
GCCTTTAAGAACTGTAACACTCACCACGAGGGTCCCCGGCTTCATTCTTGAAGTCAGTGAAACCAAGAACCACCAATTCCGGAC
ACAGTATGTCAGAAACAATATGAGTCACTAAATCAATATACTTCTCAACAATTTCCAACAGCCCTTGCAATTAACCTGGCCATGT
GACTGGTTGTGACTAAAATAATGTGGAGATAATAATGTGTTACTCCCTAAGGCAGAGTGCCCTTCTATCATTCTCTTTCCCTTCC
TCTATGTGGCAGAAAGTAAAGATTCTGAAATGATAAAGTCAATCACAGGAAGGCACCTGGACTCCTGGCCCACTGCTTGGAGGA
GAGCACTCAGGACCATGAACATCTGACTGTGACGTAGCAATAAAGAAACCCACGTTTCATATGAAACTGCTTAAATTAATGGCA
CAAGTCATGTTTTTGATGTTGCACATTTGTCTTTATTTGTGGCTTGTTTTGCTTCCACATCAATCCACTCAAGGCCTACATTCTG
CTATAATGCAATTTCAAGTTCTTTACAGGCCGAGAAAAATGAATCTGAATTCCTGACCTCCAAAAGTGATCAAGATATTTTGTAGT
TCAGGCTCCAAAATTTTCTCATTTCATAGGTTTTCTCGATTGATCATTATTCATGATTGCAAGGAATCATTCAATGTTTTCT
AAATCTATTACTGCATCCTGACACATATGACATTTTAACTATGTTCCAGATTTTTGAATGAAGAGTGTAATTTTAAATGTTTTT
ACCACAAAAAATAAGTATGTGAAGTGGTGGATTGTTAATTAGCCTTATTTAACCATTTAATATTGTACACGTACACCAAAGCAT
CATGTTGTACCCCATGAATAC--A-CACAATTATTATTGTC--AATTTAAA--AT--GAAATAATAAAAAATAACAAAGGCAT
TAGCCTCTGCATTGCCTTTACCGGTGATCCTCACGGTGACTAACGCAAAAAACGTTCTATTTTCATCCTTACAAACATCCCTATCT
TTGATGCCTCTTTGTCTAGATCTCTATCCCCTCCTGTTTTCTCTACGTTATTTATATGGGTATCA--TCACCATCC--TGGACAA
-CATCAGGAC-A-----GATATC-CCTC--A--CCA-AGCCAATG-TTCCTCTCT-ATG-TTG--GCTCAA-A-TG
TC-C-TTG-AACTTTC-----CTTTCACCAC--CCTT-T--CCACAG-----TCAAAAG-----GATATTGTAGTTTAA
TGCCTCAGAGTTCAGCTT-----T-TAAGCTTCTGA-CAAATTATCTT--CCTCTT--TAGGTT--CTC--CTTT-A
-TGG-AATCTT-----CTGT-A-----C-----TG-ATGG-CCA--TGTCTTTA-----ACTACT-ATGT
AGA-TATC--T-GCT-ACTA-CCTGTATTAT-G--CCTCTACC-TT--TATTAGCAG--AGTTA-TCTGT-ACT-GTTGGC-ATG
AC-AATCA---TTTG-----T-TAATATGAC-TTGCC-T-----TTCCTTTTCTGC-----TAT-T-CTTGATCAAA
TGGC--T---CCTC-----TTTCTTGCTCCTCTCATTCTCCTGCCTTCACTTGGACGTGCTTCACGTAGTCTG-TGCTTATG-A
CTGGATTAATAA--TTGA--T--ATGGACTTAT-C--CTAA-T-GTTGTTGCTCATAATATGGGT-----
-----TTTA-----T-GGT-CCATTATTATTTCTATGCATTGATCTGGAG-A-AGGCT--T-C-AATCC--TTT
T-A--CTCTTTGTGGAATA-----TC--TGTAACCTTCTG-GT--TCACTC--TGC-TATAGC-AATT-----T-CAG
TTTAGGCTAGT-AAGCATG-AGGATGCCTCCTTCTCTGATTTTTCCACAGTCTGTTGGTCACAGAATAACCTGAGTGA-TTACT
GATG--AAAG--AGTGAGAA-TGT-T-ATTGA-TAGTC-AC-AATGACA-AAAAACA-A-ACAAC--ACAG-TCA-----
-----AAA-T
8.558042pranav@pranav-OMEN-by-HP-Laptop:~$
```

simd parallel: 9.789405

```
pranav@pranav-OMEN-by-HP-Laptop: ~
File Edit View Search Terminal Help
GCCTTTAAGAACTGTAACACTCACCACGAGGGTCCCCGGCTTCATTCTTGAAGTCAGTGAAACCAAGAACCACCAATTCCGGAC
ACAGTATGTCAGAAACAATATGAGTCACTAAATCAATATACTTCTCAACAATTTCCAACAGCCCTTGCAATTAACCTGGCCATGT
GACTGGTTGTGACTAAAATAATGTGGAGATAATAATGTGTTACTCCCTAAGGCAGAGTGCCCTTCTATCATTCTCTTTCCCTTCC
TCTATGTGGCAGAAAGTAAAGATTCTGAAATGATAAAGTCAATCACAGGAAGGCACCTGGACTCCTGGCCCACTGCTTGGAGGA
GAGCACTCAGGACCATGAACATCTGACTGTGACGTAGCAATAAAGAAACCCACGTTTCATATGAAACTGCTTAAATTAATGGCA
CAAGTCATGTTTTTGATGTTGCACATTTGTCTTTATTTGTGGCTTGTTTTGCTTCCACATCAATCCACTCAAGGCCTACATTCTG
CTATAATGCAATTTCAAGTTCTTTACAGGCCGAGAAAAATGAATCTGAATTCCTGACCTCCAAAAGTGATCAAGATATTTTGTAGT
TCAGGCTCCAAAATTTTCTCATTTCATAGGTTTTCTCGATTGATCATTATTCATGATTGCAAGGAATCATTCAATGTTTTTCT
AAATCTATTACTGCATCCTGACACATATGACATTTTAACTATGTTCCAGATTTTTGAATGAAGAGTGTAATTTTAAATGTTTTT
ACCACAAAAAATAAGTATGTGAAGTGGTGGATTGTTAATTAGCCTTATTTAACCATTTAATATTGTACACGTACACCAAAGCAT
CATGTTGTACCCCATGAATAC--A-CACAATTATTATTGTC--AATTTAAA--AT--GAAATAATAAAAAATAACAAAGGCAT
TAGCCTCTGCATTGCCTTTACCGGTGATCCTCACGGTGACTAACGCAAAAAACGTTCTATTTTCATCCTTACAAACATCCCTATCT
TTGATGCCTCTTTGTCTAGATCTCTATCCCCTCCTGTTTTCTCTACGTTATTTATATGGGTATCA--TCACCATCC--TGGACAA
-CATCAGGAC-A-----GATATC-CCTC--A--CCA-AGCCAATG-TTCCTCTCT-ATG-TTG--GCTCAA-A-TG
TC-C-TTG-AACTTTC-----CTTTCACCAC--CCTT-T--CCACAG-----TCAAAAG-----GATATTGTAGTTTAA
TGCCTCAGAGTTCAGCTT-----T-TAAGCTTCTGA-CAAATTATCTT--CCTCTT--TAGGTT--CTC--CTTT-A
-TGG-AATCTT-----CTGT-A-----C-----TG-ATGG-CCA--TGTCTTTA-----ACTACT-ATGT
AGA-TATC--T-GCT-ACTA-CCTGTATTAT-G--CCTCTACC-TT--TATTAGCAG--AGTTA-TCTGT-ACT-GTTGGC-ATG
AC-AATCA---TTTG-----T-TAATATGAC-TTGCC-T-----TTCCTTTTCTGC-----TAT-T-CTTGATCAAA
TGGC--T---CCTC-----TTTCTTGCTCCTCTCATTCTCCTGCCTTCACTTGGACGTGCTTCACGTAGTCTG-TGCTTATG-A
CTGGATTAATAA--TTGA--T--ATGGACTTAT-C--CTAA-T-GTTGTTGCTCATAATATGGGT-----
-----TTTA-----T-GGT-CCATTATTATTTCTATGCATTGATCTGGAG-A-AGGCT--T-C-AATCC--TTT
T-A--CTCTTTGTGGAATA-----TC--TGTAACCTTCTG-GT--TCACTC--TGC-TATAGC-AATT-----T-CAG
TTTAGGCTAGT-AAGCATG-AGGATGCCTCCTTCTCTGATTTTTCCACAGTCTGTTGGTCACAGAATAACCTGAGTGA-TTACT
GATG--AAAG--AGTGAGAA-TGT-T-ATTGA-TAGTC-AC-AATGACA-AAAAACA-A-ACAAC--ACAG-TCA-----
-----AAA-T
8.003543pranav@pranav-OMEN-by-HP-Laptop:~$
```

SPEEDUP:

DATA-SET	5000	8000	10000
omp parallel for	1.01274946	1.06107805	1.01596835
simd parallel	1.07322457	1.09233372	1.02769504



CONCLUSION:

In our project, we have implemented the Smith-Waterman Algorithm in a serial and parallel manner (openMP, openMP+SIMD) for local alignment of gene sequences. We have compared the results of the implementation for sequential, parallel using openMP, parallel using openMP+SIMD. Analysis have been carried out after comparisons and results have been presented. At last we would like to conclude that parallel implementation of local gene sequencing using Smith-Waterman algorithm is advantageous when compared to sequential one. With the development in biotechnology parallel implementation would actually help in processing out large data-sets used for this problem.

ACKNOWLEDGMENT:

Although we have taken efforts in this project, this project would not have been possible without the kind support and help of some individuals and organizations. I would like to extend my sincere thanks to all of them.

We are highly indebted to Prof. Dr. SAIRABANU J for her guidance, constant supervision and also for providing necessary information regarding the project.

We would also like to extend our deepest gratitude to all those who have directly and indirectly guided us in completing this project. Many people, especially our classmates and team members itself, have made valuable comment suggestions on this proposal which gave us an inspiration to improve our work. We thank all the people for their help directly and indirectly to complete our project.

My sincere thanks for our University and our School of Computing Sciences and Engineering Department for allowing us to carry out this project.

PRANAV MANDALA 17BCE0803

MOHITH J 17BCE2135

RASHI KASERA 17BCE2421

REFERENCES:

1. M.P Sudha, P.Sripriya ;(2006) Sequence Alignment in DNA Using Smith Waterman and Needleman Algorithms;

<http://ijcsit.com/docs/Volume%205/vol5issue04/ijcsit20140504244.pdf>

2. FN Muhamad, RB Ahmad, SM Asi, MN Murad;(2018) Performance Analysis Of Needleman-Wunsch Algorithm (Global) And Smith-Waterman Algorithm (Local) In Reducing Search Space And Time For Dna Sequence Alignment;

<https://iopscience.iop.org/article/10.1088/17426596/1019/1/012085>

3. Kartika Ahuja, Kompal;(2015) Analysing Multiple DNA Sequence Alignment Algorithms- Smith Waterman Algorithm and Parallel Smith Waterman Algorithm;

<https://www.ijraset.com/files/serve.php?FID=379>

4. Deepa. B. C , Nagaveni. V;(2015) Parallel Smith-Waterman Algorithm for Gene Sequencing

<https://pdfs.semanticscholar.org/09eb/71e3c630f399dd24f9f0ddfd6a624effa54f3.pdf>

5. Liang-Tsung Huang, Chao-Chin Wu, Lien-Fu Lai and Yun-Ju Li;(June 2015) Improving the Mapping of Smith-Waterman Sequence Database Searches onto CUDA-Enabled GPUs

<https://www.hindawi.com/journals/bmri/2015/185179/>

6. Yanfeng Liu, Leixiao Li, Jing Gao; (March 2019) An Improved Smith-Waterman Algorithm Based on Spark Parallelization;

https://www.researchgate.net/publication/332075992_An_Improved_SmithWaterman_Algorithm_Based_on_Spark_Parallelization

7. Yu Liu, Yang Hong, Chun-Yuan Lin, and Che-Lun Hung;(October 2019) Accelerating Smith-Waterman Alignment for Protein Database Search Using Frequency Distance Filtration Scheme Based on CPU-GPU Collaborative System;

<https://www.ncbi.nlm.nih.gov/pubmed/26568953>

8. Amadou Chaibou, Oumarou Sie;(June 2015) Comparative Study of the Parallelization of the Smith-Waterman Algorithm on OpenMP and Cuda C;

https://www.researchgate.net/publication/281431935_Comparative_Study_of_the_Parallelization_of_the_SmithWaterman_Algorithm_on_OpenMP_and_Cuda_C

9. Enzo Rucci, Carlos Garcia Sanchez;(April 2019) SWIMM 2.0: Enhanced Smith– Waterman on Intel’s Multicore and Manycore Architectures Based on AVX-512 Vector Extensions;

<https://link.springer.com/article/10.1007/s10766-018-0585-7>

10. Matthew Baker, Aaron Welch, Manjunath Gorentla Venkata;(January 2015) Parallelizing the Smith-Waterman Algorithm using Open SHMEM and MPI-3 One-Sided Interfaces;

https://link.springer.com/chapter/10.1007/978-3-319-26428-8_12

11. Zeiad El-Saghir, Hamdy Kelash, SayedElnazly, HossamFaheem;(December 2016) Parallel Implementation of Smith-Waterman Algorithm using MPI, OpenMP and Hybrid Model;

<https://www.semanticscholar.org/paper/Parallel-Implementation-of-SmithWaterman-Algorithm-El-SaghirKelash/926619d67aa05cd9a7153a02d441f289969cf6f4>

12. Enzo Rucci, Carlos Garcia, Guillermo Botella, Armando De Giusti, Marcelo Naiouf & Manuel Prieto-Matias; (November 2018) SWIFOLD: Smith-Waterman implementation on FPGA with OpenCL for long DNA sequences;

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6245597/>

13. Rahulkumar Gayatri, Charlene Yang, Thorsten Kurth, Jack Deslippe;(2015) A Case Study for Performance Portability using OpenMP 4.5

https://link.springer.com/chapter/10.1007/978-3-030-12274-4_4

14. Chih-Hung Chang, Chih-Wei Lu, Chao-Tung Yang, Tzu-Chieh Chang; (April 2016) An approach of performance comparisons with OpenMP and CUDA parallel programming on multicore systems;

https://www.researchgate.net/publication/299591892_An_approach_of_performance_comparisons_with_OpenMP_and_CUDA_parallel_programming_on_multicore_systems

15. Zeiad El-Saghir, Hamdy Kelash, SayedElnazly, HossamFaheem;(December 2016) Parallel Implementation of Smith-Waterman Algorithm using MPI, OpenMP and Hybrid Model;

<https://www.semanticscholar.org/paper/Parallel-Implementation-of-SmithWaterman-Algorithm-El-SaghirKelash/926619d67aa05cd9a7153a02d441f289969cf6f4>