Advances in Computational Sciences and Technology
ISSN 0973-6107 Volume 10, Number 8 (2017) pp. 2587-2601
© Research India Publications
http://www.ripublication.com

Study and Analysis of Various Bioinformatics Applications using Protein BLAST: An Overview

Sita Rani
Ph.D. Research Scholar, I.K.G. Punjab Technical University, Kapurthala, Punjab, India.

OP Gupta
Associate Professor, School of Electrical Engineering & Information Technology, Punjab Agricultural University, Ludhiana, Punjab, INDIA.

Abstract

Bioinformatics is a multidisciplinary field. It is the application of information technology and statistics in biology. In bioinformatics, computers are used to store, analyze and interpret a huge volume of biological data. Various application areas of bioinformatics include sequence analysis, genome annotation, analysis of gene expressions, comparative genomics, protein structure prediction etc. Among these areas sequence analysis acts as fundamental step in many research applications. And key activity performed in analysis is sequence alignment which contributes to uncover genetic evolutions of various organisms.

In this paper various biological databases and methods of sequence alignment are discussed. As BLAST is the most frequently used sequence alignment algorithm, so BLAST is reviewed in detail. Various accelerated versions of protein BLAST are also studied and discussed. At the end, it is concluded that because of huge volume of biological data available in various biological databases, any improvement in the execution time of BLAST can further benefit many bioinformatics applications.

Keywords: Bioinformatics; BLAST; Databases ; Data Analysis; Molecular Biology; Sequence Alignment; System Biology;

INTRODUCTION

An unbelievable transformation has been recognized in science with recent technological approaches, which have generated a huge volume of biological data. This data is available in various public databases and is a big challenge for scientists from various research areas [1]. The main challenge is to decode this huge volume of sequential and structural data that have been evolved from various biological processes. In bioinformatics, various tools are used to organize, analyze and interpret the data [2].

A new age of "biology" has evolved in collaboration with many other areas of science such as computational biology and bioinformatics. Bioinformatics has come up with a magnificent effect on the analysis and interpretation of data available in the various databases.

The main objective of this study is to present a brief overview of bioinformatics, various biological databases and bioinformatics applications. Basic Local Alignment Search Tool (BLAST) is one of the the most frequently used algorithm for various bioinformatics activities, so a detailed study of BLAST is presented in this paper. Various accelerated implementations of protein BLAST are also discussed. Further the scope of improvement in the execution time of protein BLAST is also discussed.

WHAT IS BIOINFORMATICS?

Bioinformatics came into existence with the inception of DNA sequencing [3]. The development responsible for its emergence is the publication of the structure of DNA by Watson and Crick in 1953 besides the accumulation of data and knowledge of biochemistry and protein structure with the studies of Pauling, Coren, and Ramachandran in the 1960s [4].

Margaret O. Dayhoff is assumed the mother of bioinformatics for his contribution in the organization of data and knowledge of 3-D protein structure [5]. Computers able to identify the peptide sequence, software programs to identify and show structures for use in X-ray crystallography and many other computational methods were developed by Margaret O. Dayhoff [3] and [4].

Apart from this many other scientists, engineers and researchers have contributed to the development of numerous bioinformatics applications till now, which would not have been possible without the evolution of the computer. Thus, the compelling developments made in the field of bioinformatics are because of technological advances in the computational power and various genome projects (annotation, sequencing, analysis of data and processing) [4].

However, with the repeated experiments in the field of next-generation sequencing (NGS), the list of complete genomes is multiplying, as well as the volume of data. So, it becomes unavoidable to use computers in the field of molecular biology to perform various activities related to storage, analysis and processing of data to identify the functional and structural features of newly generated genome sequences [1].

Bioinformatics has a multifaceted character, so this can be characterized as “the application of computational tools to organize, analyze, understand, visualize and store information associated with biological macromolecules” [6] and [2]. The field of bioinformatics can also be epitomized as: i) the cell and the central dogma of molecular biology ii) the organism, which shows changes between the different stages of development and regions of the body iii) the tree of life, in which millions of species are grouped into three evolutionary branches [2]. Bioinformatics has applications in various dimensions like sequence alignment, phylogenetics, system biology, homology modeling etc. [4]. Some of the bioinformatics applications are shown in Figure 1.

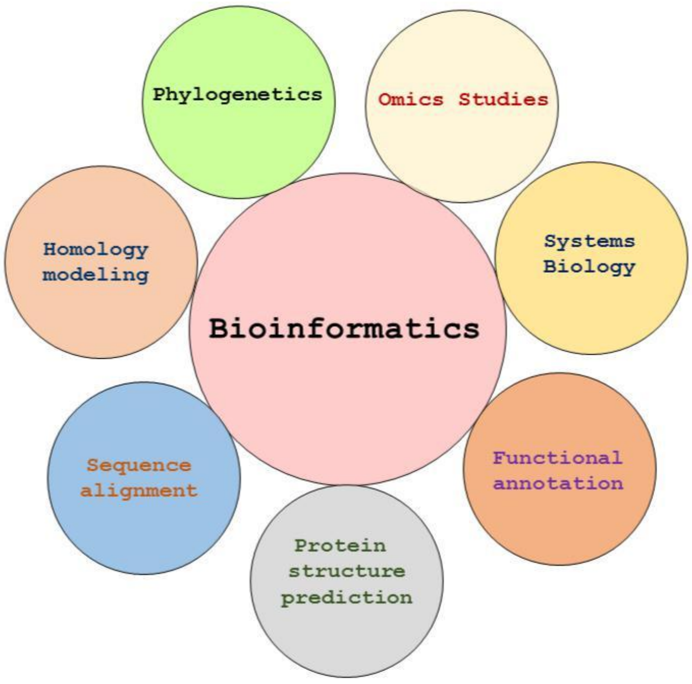


Figure 1. Bioinformatics Applications [4]

A detailed aspect of the field is portrayed by Luscombe et al. [6]. These authors explained the three main objectives of bioinformatics: i) to store the data efficiently so that scientists and researchers can easily access the data; ii) to design and develop tools for the analysis of the data. and iii) to use these tools to interpret the data accurately.

BIOLGICAL DATABASES

An enormous amount of biological data has been generated from various bioinformatics applications. Its storage and organization has become a very important issue. Many biological databases have been generated and maintained to provide easy access to the researchers and scientific community to the stored information [6] and [7]. As the volume of biological data is increasing from day to day experiments performed in the field, so number of biological databases is also increasing. These databases also need to be maintained and updated time to time. As reported in the most recent published facts, there are 1739 biological databases. Data stored in these databases can be categorized as raw DNA sequences, protein sequences, macromolecular structures, genome sequencing etc.

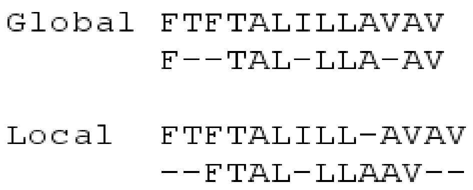
Databases are also categorized as primary and secondary databases. The primary databases store results of experimental data that are published without careful analysis related to previous publications. Whereas, secondary databases are used to store compilation and interpretation of data, called content curation process [7]. Besides these, there are functional databases such as the Kyoto Encyclopedia of Genes and Genomes (KEGG) and Reactome that allow analysis and interpretation of metabolic maps [9]. Few examples of primary databases are GenBank at the National Center for Biotechnology Information (NCBI), DNA Database of Japan (DDBJ), and European Molecular Biology Laboratory (EMBL) stand out as the main databases of nucleotide sequences and proteins [2]. These databases are members of the International Nucleotide Sequence Database Collaboration (INSDC) and share among each other the deposited information daily [8]. Some examples of secondary databases are Protein Information Resource (PIR), UniProtKB/Swiss-Prot, Protein Data Bank (PDB), Structural Classification of Proteins (SCOP), and Prosite. These databases are curated and present only information related to proteins, describing aspects of its structure, domains, function and classification.

GenBank is the most accessed and known throughout the world public database [2], with over 198,565,475 million sequences deposited (release 217, December 2016). Given the enormous amount of molecular data, they are categorized according to their nature (DNA, RNA, protein...) and are used, among other applications, in the analysis of sequence comparison [9].

SEQUENCE ALIGNMENT

In bioinformatics, comparison among the molecular sequences to identify regions of similarity is called sequence alignment. It is a key step in data processing in many bioinformatics applications. Its basic purpose is to reveal functional, structural or evolutionary relationships in different sequences.

In sequence alignment, regions in different sequences with maximum match are identified with some algorithms. Output of the algorithm is analyzed to judge the relationship between the biological properties of the compared sequences. There are two different well known techniques for sequence alignment i.e. Global Alignment and Local Alignment as shown in Figure 2 [10]. In Global Alignment every element of query sequence is aligned with compared sequence/sequences. This technique is preferred when sequences are almost of same length and similar. But when the sequences to be aligned are of different lengths and are dis-similar but are expected to contain some regions of similarity, Local Alignment is used. Sequence alignment can further be categorized as pair wise or multiple. Pair wise Sequence Alignment is used to identify similarities between two sequences at a time. In Multiple Sequence Alignment more than two sequences can be considered at same time for best possible alignment.



```
Global  FTFTALILLAVAV
        F--TAL-LLA-AV

Local   FTFTALILL-AVAV
        --FTAL-LLAAV--
```

Figure 2. Global and Local Sequence Alignment

Different bioinformatics algorithms are available for different types of sequence alignment. A well known global alignment method is Needleman-Wunsch algorithm and famous local alignment methods are Smith-Waterman algorithm and BLAST (Basic Local Alignment Search Tool). Depending upon type of application one or another algorithm is used for sequence analysis [11], [12] and [13].

BLAST

BLAST is the most commonly used algorithm to execute queries on biological databases. When a query sequence is given, the main objective is to find similar sequences from the available databases. Similar sequences can also be searched by calculating the pair wise score of the alignments between the query sequence and the database sequences, as it is done in the Smith – Waterman algorithm. But in this computational cost is very high. BLAST uses some heuristic methods to reduce the running time with little sacrifice in accuracy [14]. BLAST is used to identify the regions of local similarity between query sequence and database sequences by considering some threshold value. For BLAST algorithm, input is given in FASTA format. FASTA format is represented as: >gi|5524211|gb|AAD44166.1| cytochrome b

```
LCLYTHIGRNIYYGSYLYSETWNTGIMLLLITMATAFMGYVLPWGQMSFWG
ATVITNLFSAIPYIGTNLVEWIWGGFSVDKATLNRFFAFHFILPFTMVALAGVH
LTFLHETGSNNPLGLTSDSDKIPFHPYYTIKDFLGLLILILLLLLLALLSPDMLG
DPDNHMPADPLNTPLHIKPEWYFLFAYAILRSVPNKLGGVLALFLSIVILGLMP
FLHTSKHRSMMLRPLSQALFWTLTMDLLTLTWIGSQPVEYPYTIIGQMASILY
FSIILAFLPIAGXIENY
```

Where ">" sign acts as sequence identifier and remaining text gives the description. So both the query sequence and database sequence need to be converted into FASTA format before applying the algorithm. BLAST's final result consists of a series of local alignments, ordered by the similarity score achieved in the extension process. It presents the complete sequences matched, score, size of the alignment and also a value based on the probability that the alignment has been found by chance, the e-value. The e-value depends on the query and database composition, and some of the alignments with higher e-values are discarded from the result. Various tasks performed for this search are executed in four different stages of protein BLAST, discussed below [11]:

- **Hit Detection:** Here, the comparison is done between the query sequence and database sequences to locate words matches for specified word length and each identified word match is called a hit.
- **Un-gapped Extension:** In this stage, the score for each word match is compared against the threshold value after extending the word match in both the directions without gaps. Matches, which score higher than the threshold value are called High Scoring Pairs (HSPs), are considered for next stages and all other word matches are discarded.
- **Gapped Alignment:** In this stage, un-gapped alignments retained in the previous stage are extended by inserting gaps. Alignments with a score higher than the threshold value are called High Score Alignments (HSAs).
- **Gapped Alignment with Trace- back:** Here trace-back path is determined for the final alignments and results are displayed to the user.

BLAST Parameters

Some important parameters of protein BLAST are discussed below [13]:

- **Word size (w):** It specifies the length of the match. This parameter is defined by the user. The algorithm is executed with specified word length to find the matches in the sequences. The smaller the value of w, more the number of matches will be identified during the alignment and vice-versa.

- **Threshold (t)**: It is also a user defined parameter. It is the minimum defined value for a word match. Matches with score less than the threshold are assumed to occur by chance and discarded. Lesser the threshold value, there is probability of higher number of hits.
- **Drop-off (x):** Score in an alignment, is permitted to fall by drop-off value as compare to already available highest score value.
- **Lambda (ƛ):** ƛ is matrix specific constant. The score is normalized using matrix specific constant.
- **Adjustment (k):** It is called Adjustment factor. It is assumed that alignments at different locations and their scores may be co-related.
- **Gap penalty:** Gap penalty reduces the effect of idles in an alignment.

BLAST Variants

Because of the variety of the biological sequences to be worked upon, different variants of BLAST are available and shown in Table 1 [15]. These are:

- **Blastn:** In Blastn, nucleotide queries are executed on nucleotide databases for alignment. In this variant of BLAST, short sequences of nucleotides also called oligonucleotides are compared with long sequences to identify characteristic of unknown nucleotide sequences.
- **Blastp:** In Blastp, both query sequences and database sequences are proteins. In this a query sequence of unknown features is compared with protein sequences of known functionality in the databases. Functionality of the unknown sequence is identified from best aligned sequences from the database.
- **Blastx:** In Blastx, a nucleotide sequence is aligned with protein databases. Before the alignment nucleotide query sequence is converted to protein sequence. Three protein sequences are generated for each nucleotide sequence. The first sequence is obtained by converting three nucleotides to a protein. Then another two protein sequences are generated in the same method, but leaving first and then first two letters of the original nucleotide sequence. The Blastx is mainly used in genomic DNA to identify protein coding genes.

Table 1: BLAST variants, type of query and database sequences

BLAST Variant	Type of Database	Type of Query Sequence
Blastn	Nucleotide	Nucleotide
Blastp	Protein	Protein
Blastx	Protein	Translated Nucleotide
tBlastn	Translated Nucleotide	Protein
tBlastx	Translated Nucleotide	Translated Nucleotide

- **tBlastn:** In tBlastn, a protein query sequence is executed on a nucleotide database. To obtain accurate alignments, before query execution, each database sequence is converted to three protein sequences as explained above in Blastx. Then query sequence is aligned with different protein sequences in resultant database. Its application is in protein mapping of the genome.
- **tBlastx:** In tBlastx, both query sequence and databases, both are of nucleotide type. So both query sequence and database sequences are converted to protein sequences first (as explained above), then query sequence is processed in the database. tBlastx is used to determine transcripts of unknown functions.

BIOINFORMATICS APPLICATIONS

Bioinformatics is a multidisciplinary field. It is the application of information technology and statistics in biology [16], [17] and [18]. In bioinformatics computers are used to store, analyze, manipulate and distribute biological data such as protein sequences, DNA or RNA [19] and [20]. Various application areas of bioinformatics include sequence analysis, genome annotation, analysis of gene expressions, comparative genomics, protein structure prediction etc [21]. Among these areas sequence analysis acts as fundamental step in many bioinformatics applications. And key activity performed in analysis is sequence alignment which contributes to uncover genetic evolutions of various organisms.

BLAST is the most frequently used tool in various bioinformatics applications to perform the task of sequence alignment. This algorithm is based on heuristic approach and is much faster than other approaches. The emphasis of calculating an optimal alignment on speed is vital and to make the algorithm practical on the huge genome databases which are currently available. Although the subsequent algorithms can be even faster, BLAST is most widely used algorithm. Many attempts have been made, since the time of its introduction to design and develop new accelerated BLAST software tools for specific hardware platforms [22], [23], [24] and [25].

BLAST was first time introduced in 1990. Initially BLAST was designed only for un-gapped alignments [14]. A revised version of BLAST for gapped alignments was proposed in 1997. Gapped BLAST had many advantages as compare to original BLAST. It was approximately three times faster than original BLAST and was more sensitive. In PSI-BLAST, statistically more relevant alignments were added to the position-specific score matrix automatically and search was performed using this matrix. There were many statistical improvements in new proposed BLAST [26] .

Another implementation of BLAST was proposed for LINUX cluster by splitting the database on different nodes of the cluster i.e. mpiBLAST . Message Passing Interface (MPI) was used for communication among nodes. In this implementation greedy approach was implemented at the server to allocate different database fragments to the nodes. As a smaller portion of the database was available on each node and all the nodes were performing search process in parallel, so it was faster than the simple implementation. It was shown experimentally, when blast executed on multiple nodes a superliner speedup was obtained [27]. Many other versions of modified BLAST were proposed over the time for different environments, to accelerate the process of sequence alignment in bioinformatics applications.

With the embodiment of graphic processing units (GPUs) in general purpose computing, a new era of bioinformatics applications started. CUDA is a hardware and software platform provided by NVIDIA, which is specially designed to explore the parallel architecture of GPUs by using multithreaded code [28]. In most of the bioinformatics applications, there is a need to deal with huge databases, so these applications are data intensive [29], [30], [31] and [33]. By using the multithreaded programming approach of CUDA on GPUs, these applications can be accelerated greatly. Many bioinformatics algorithms and tools have already been optimized on GPUs. Some authors have been already proposed accelerated GPU- enabled implementations of BLAST [32].

In the very first GPU implementation of protein BLAST by Ling et al., the authors used two-hit method for gapped-BLAST. Two kernels were designed in this implementation. First kernel was used to perform the steps up to un-gapped alignment and last two steps were implemented by second kernel. It was discussed that seeding stage of BLAST plays main role in the performance of a GPU implementation of BLAST. When there are fewer matches identified, lesser number of threads is executed in parallel. GPU memory is utilized to store different data structures. Experiments were performed by the authors against Swiss-Prot protein database. The authors claimed a speedup of 1.7X to 2.7X as compare to standard NCBI- BLAST. It was also highlighted that for smaller query sequences, speedup achieved was high as compare to longer query sequences [34].

In another implementation by Xiao et al., it was demonstrated that 99% of the execution time is consumed by first three stages of BLASTP. So authors focused on these stages in their implementation. They also explained that the first two stages cannot be isolated from each other. In this implementation for hit detection, words were picked from the database sequence one by one and compared with query sequence. In the next step, multithreading was used to align sequences, where each thread aligned a query sequence and a database sequence. In their implementation, they also optimized the performance of the algorithm by placing different data structures in suitable memories of the GPU. Scoring matrix and query sequence were placed in constant memory because of the fast access time and small size of these structures. Subject sequence and word lookup table were stored in texture memory because they are large in size and faster to access. In this implementation, sequences were assigned to different threads using a greedy algorithm. Parallelization was done only for first two stages i.e. hit detection and un-gapped extension. They demonstrated with their experiments that when first two phases were implemented on the GPU, a speedup of 7X was obtained for these two phases only. But when both, execution of Dual core CPU and GPU was pipelined then a speedup of 6X was obtained in total execution time [33]. To provide these speedup figures, following five different implementations were tested by the authors:

- Serial execution on CPU.
- The GPU was used to implement all the stages (three).
- The GPU was used for first and second stage and the third stage was implemented on the CPU.
- The GPU was used for first and second stage and third stage executed on two threaded CPU
- The GPU is used for first and second stage in parallel to two threaded CPU for the third stage.

Liu et al., gave another GPU implementation of BLAST for protein databases. In this implementation, the authors focused on data structure design. Hit detection information was stored with compressed Deterministic Finite State Automaton (DFA). The GPU was used by the authors to implement stage 1, 2 and 3, whereas stage 4 was implemented on the CPU. Two different kernels were designed for the GPU implementation of this approach. Stage 1 and 2 were implemented with one kernel and a second kernel for stage 3.During algorithm execution, each thread worked on one database sequence to identify word matches. For each word match, first un-gapped extension was performed by taking threshold value. Then gapped extension was performed by the GPU. Final results were compiled using trace-back by the CPU in stage 4 and are displayed. This implementation was memory optimized and all the data structures were placed in best possible memories by considering the size of data structure and memory and frequency of access. The authors claimed a speedup of 10X as compared to standard NCBI-BLAST. But alignments generated did not match with standard NCBI-BLAST, so there is a doubt for its use by other researchers [36].

GPU-BLAST by Vouzis et al., is the most reliable among all the discussed implementations because this was designed on the top of standard NCBI-BLAST's source code. Input and output format were also same. Alignments generated exactly matched with NCBI-BLAST. The authors explained that the majority of the data structures used in their implementation were the same as NCBI-BLAST. Only some optimization was done to exploit the GPU. Along with GPU, even multi core CPU was also utilized to its maximum level. And load was well distributed between the CPU and GPU, so that when GPU is in execution CPU should not be idle. GPU's global, constant and shared memories were used in this design. Authors have told that local and texture memories were not being used in their approach. Some of the important data structures used by the authors in this implementation are:

- Substitution matrix
- Presence bit vector
- Query index table
- Overflow table

These data structures were also optimized by considering the size and frequency of usage. In this implementation, GPU and CPU both were used during seeding and extension phases in parallel. After calculation of High Scoring Pairs by the CPU and GPU, the results were given back to the CPU and the rest of the execution was performed by the CPU as done in standard NCBI-BLAST. The authors claimed a speedup between 3X and 4X in comparison to standard NCBI-BLAST. There is a special feature of this implementation; both the options are available for CPU BLAST as well as for GPU BLAST. This feature is not provided in any previous implementation [37].

Apart from protein BLAST, many other bioinformatics applications have also been benefitted and accelerated by GPUs. A GPU accelerated design of nucleotide BLAST is also proposed by Zhao et al. [2014]. This implementation provided a speed up, ranging from 7.15X to 14.80X when executed under two different modes i.e. megaBLAST and multithreaded respectively [38].

CONSIDERATIONS AND PERSPECTIVES

In bioinformatics applications, a huge volume of data is to be processed to perform various tasks. So processing time is always a very important parameter to be considered. Even in all the BLAST implementations discussed above, authors have evaluated the performance of their proposed version of BLAST by analyzing the execution time and speed-ups in comparison to existing BLAST implementations. Although many accelerated implementations of protein BLAST have been already proposed by various authors, but as the volume of biological data is growing day-by-day, execution time of BLAST further need to be improved to speed-up various bioinformatics applications. As many GPU implementations of protein BLAST with considerable speed-up in comparison to standard NCBI-BLAST are also available, protein BLAST can further be improved (in terms of execution time) by implementing it on GPU-enabled high performance cluster. In Parallel implementations of the algorithm communication delay and bandwidth of the communication media also play very important role.

ACKNOWLEDGEMENT

The authors express deep gratitude to the Dean, Research, Innovation and Consultancy Department of I.K.G. Punjab Technical University, Kapurthala for giving them the opportunity to carry on this research work.

REFERENCES

1. Ritchie MD, Holzinger ER, Li R, Pendergrass SA, et al. (2015). Methods of integrating data to uncover genotype-phenotype interactions. Nat. Rev. Genet. 16: 85-97.
2. Pevsner J (2015). Bioinformatics and functional genomics, 3rd ed. John Wiley & Sons Inc, Chichester.
3. Hagen JB (2000). The origins of bioinformatics. Nat. Rev. Genet. 1: 231-236.
4. Verli H (2014). O que é Bioinformática? In: Bioinformática da biologia à flexibilidade molecular (Verli H ed.). SBBq, São Paulo, 1-12.
5. Hunt LT (1984). Margaret Oakley Dayhoff 1925-1983. Bull. Math. Biol. 46: 467-472.
6. Luscombe NM, Greenbaum D and Gerstein M (2001). What is bioinformatics? A proposed definition and overview of the field. Methods Inf. Med.
7. Prosdocimi F (2010). Introdução à bioinformática. Curso Online. Available at [http://www2.bioqmed.ufrj.br/prosdocimi/ FProsdocimi07_CursoBioinfo.pdf].
8. Prosdocimi F, Cerqueira GC, Binneck E and Silva AF (2002). Bioinformática:

Manual do usuário. Biotec. Cienc. Des. 12-25.

9. Amaral AM, Reis MS and Silva FR (2007). O programa BLAST: guia prático de utilização. 1st edn. Embrapa Recursos Genéticos e Biotecnologia. EMBRAPA, Brasília.

10. Gupta OP and Rani S (2013). Accelerating Molecular Sequence Alignment using Distributed Computing Environment, International journal of scientific and Engineering Research. 10:4, 262-265.

11. Vinh LV, Lang TV, Du NT and Chau VH (2012). Multiple sequence Alignment using Grid Computing using Cache Technique, International Journal of Computer Science and Telecommunication. 3 : 7, 46-51.

12. Mathkour H and Ahmed M (2010). A Comprehensive Survey on Genome sequence Analysis, IEEE International Conference on Bioinformatics and Biomedical Technology. 14-18.

13. Chen Y, Yu S and Leng M (2006). Paralleln sequence Alignment Algorithm for Clustering System, International federation for Information Processing. 311-321.

14. Altschul SF, Warren G, Miller W, Eugene WM and Lipman DJ (1990). Basic Local Alignment search Tool, Journal of Molecular Biology. 215: 3, 403-410.

15. Barton GJ (2008). Sequence alignment for Molecular Replacement, Biological Crystallography. 9 : , 2, 25-32.

16. Feng W. (2009). Massively Parallel Sequence Alignment: When Computational Biology Becomes I/O Biology, Populations, Communities, Ecosystems and Evolutionary Dynamics: Genomics and Metagenomics. 10: 4, 19-22.

17. Mcmahon PL (2008). Accelrating Genome Sequence Alignment using High Performance Reconfigurable Computer, Ph.D Dessertation, University of Cape Town.

18. Diaz D, Esteban FJ, Hernandez P, Cabbalero JA, Dorado G and Galvez S (2011). Parallelizing and optimizing a bioinformatics sequence alignmnet for many-core architecture, Parallel Computing.. 11, 244-259.

19. Wirawan A, Kwoh CK, Hieu N and Schmidt B (2008). CBESW: Sequence Alignmnet on Playstation 3, BMC Bioinformatics. 1-10.

20. Barton GJ (2008). Sequence alignment for Molecular Replacement, Biological Crystallography. 9: 2, 25-32.

21. Chen S, Lin C and Chen S (2005). Multiple DNA Sequence Alignment Based on Genetic Algorithms and Divide-and-Conquer Techniques, International Journal of Applied science and Engineering. 3:2,89-100.

22. Fei X (2008). FPGA-based accelerators for BLAST families with multi-seeds detection and parallel extension. In: Proceedings of the 2nd International

Conference in Bioinformatics and Biomedical Engineering. IEEE, Shanghai, China, 58–62.

23. Jacob A (2007.) FPGA-accelerated seed generation in Mercury BLASTP. In: Proceedings of 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines. IEEE, California, USA, 95–106.
24. Sotiriades E and Dollas A (2007). A general reconfigurable architecture for the BLAST algorithm. J. VLSI Signal Process. 48, 189–200.
25. Zhang Z (2000). A greedy algorithm for aligning DNA sequences. J. Computational Biology, 7, 203–214.
26. Lin H (2008). Massively parallel genomic sequence search on the Blue Gene/P architecture. In: Proceedings of the ACM/IEEE Conference on Supercomputing. ACM/IEEE, Austin, USA, 1–11.
27. Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W and Lipman DJ (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs, J. Nucleic Acid Research, 25(17), 3389-3402.
28. Kindratenko VV, Enos JJ, Shi G, Showerman MT, Arnold GW, Stone JE, Phillips JC and Hwu W (2009). GPU Clusters for high performance computing, IEEE International Conference on Cluster Computing and Workshop, New Orleans, LA, 1-8.
29. Fenstermacher D (2005). Introduction to Bioinformatics, J. of American Society for International Science and Technology, 56:5, 440-446.
30. Albayraktaroglu K, Jaleel A, Wu X, Franklin M, Jacob B, Tseng CW and Yeung D (2005). BioBench: A Benchmark Suite of Bioinformatics Applications, Proc. IEEE International Symposium on Performance Analysis of Systems and Soft- wares, Austin, TX, 2-9.
31. Cohen J (2004). Bioinformatics: An Introduction to Computer Scientists, ACM J. Computing Surveys, 36:2, 122-158.
32. Rani S and Gupta OP (2016). Empirical Analysis and Performance Evaluation of various GPU implementations of Protein BLAST, International Journal of Computer Applications. 151:5, 22-27.
33. Lin C, Hung C and Huang J (2013). Efficient GPU-Based Algorithm for Aligning Huge Sequence Database, IEEE International conference on High Performance Computing and Communications, Zhangjiajie, 1758-1762.
34. Ling C and Benkrid K (2010). Design and implementation of a CUDA-compatible GPU-based Core for gapped BLAST algorithm, International Conference on Computational Science is available on Science Direct Procedia

Computer Science. 1:1, 495-504.

35. Xiao S, Lin H and Feng W (2011). Accelerating Protein Sequence Search in Heterogeneous Systems, IEEE Parallel and Distributed Processing Symposium, Anchorage, AK, 112-1222.
36. Liu W, Schmidt B and Muller-Witting W (2011). CUDA-BLASTP: accelerating BLASTP on CUDA-enabled graphics hardware, IEEE/ACM Transaction on Computational Biology and Bioinformatics, 8:6, 1678-1684.
37. Vouzis PD and Sahinidis NV (2011). GPU-BLAST: using graphics processors to accelerate protein sequence alignment, BIOINFORMATICS, 27:2,182-188.
38. Zhao G and Chu X (2014). G-BLASTN: accelerating nucleotide alignment by graphics processors, J. BIOINFORMATICS. 30:10, 1384-1391.

Sita Rani is pursuing her Ph.D in Computer Science and Engineering from I.K.G. Punjab Technical University, Kapurthala. She received her B.Tech (Computer Science and Engineering) and M.Tech (Computer Science and Engineering) in the years 2002 and 2008 respectively from Guru Nanak Dev Engineering College, Ludhiana. Currently, she is working as Associate Professor cum Head of Department at Ludhiana Group of Colleges, Chaukimann. Her current research interests are Software Engineering, Data Structures, Parallel Computing and Bioinformatics.

OP Gupta, Ph.D (Computer Science & Engineering) is an alumni of PAU, Ludhiana, Thapar University, Patiala and GNDU, Amritsar has demonstrated his intellectual, interpersonal and managerial skills in various domains. He is the winner of PAU Meritorious Teacher Award for 2009-2010. Having vast industrial experience of working in IT industry with the role of Project Manager, currently he is an Associate Professor of Computer Science and Deputy Director, School of Information Technology at PAU, Ludhiana. He is also approved Ph.D supervisor of I.K.G. Punjab Technical University, Kapurthala. His areas of interests include Parallel and Distributed Computing, Grid Computing, Bioinformatics, Network Testing and Network Management.