Impact of Parallel Computing on Bioinformatics Algorithms

O.P. Gupta*, Sita Rani** and Dhruv Chander Pant***

*Associate Professor, School of Information Technology, PAU Ludhiana, Punjab, India
E-mail: opgupta@pau.edu

**Associate Professor, RIMT – IET, Mandi Gobindgarh, Punjab, India
E-mail: sitasaini80@yahoo.in

***Ph.D. Scholar, PTU Jalandhar, Punjab, India
E-mail: dpant9@gmail.com

Abstract

The biggest challenge bioinformatics facing today is to manage, analyze and process voluminous genome data. Such analysis and processing is very impractical with the help of uniprocessor computers, so the need of parallel computing in bioinformatics arises. Now distributed computers, cloud computers and multicore processors are available at very low cost to deal with bulk amount of genome data. Along with these technological developments in distributed computing, many efforts are being done by the scientists and bioinformaticians to parallelize and implement the algorithms to take the maximum advantage of the additional computational power.

In this paper various parallel computing architectures and parallel implementation of the bioinformatics algorithms are discussed. The performance analysis of the parallelized algorithms is also analyzed.

Keywords: Algorithms, Bioinformatics, Genome data, Parallel Computing,

Introduction

Bioinformatics

Bioinformatics is the applications of computer science to store, manage, analyze and process biological data [1], [2]. Bioinformatics is applied in various areas like molecular medicine, personalized medicine, preventative medicine, gene therapy, drug development, waste cleanup, climate change studies, alternative energy sources, biotechnology, antibiotic resistance, forensic analysis of microbes, bio-weapon creation, crop improvement, insect resistance, veterinary sciences etc. [3]. In all these application areas bioinformatics algorithms deal with bulk amount of genome data. Generally the bioinformatics applications face the following challenges [4]:

- To manage and process the bulk amount of genome data.
- To reduce data analysis time.

So break through technological development was needed to solve many critical problems of bioinformatics [5]. Such data management is impractical with the help of uniprocessor computers. So the use of parallel computing in bioinformatics applications is important. Now to deal with bulk amount of genome data distributed computers, cloud computers and multicore processors are also available at very low cost.

Parallel Computing Architectures

In parallel computing a problem is broken into discrete parts and instructions of different parts run on different CPUs concurrently as shown in Fig. 1.

In parallel computation a computing resource may be a single computer with multiprocessors, different number of computers connected by a network, multi core processors or the combination. And the problem should be able to be broken into different parts that can run simultaneously [6]. The various advantages of using parallel computing are:

- Save time and/or money
- Solve larger problems
- Provide concurrency
- Use of non local resources

Parallel computer systems can be classified into two main models: Single Instruction Multiple Data (SIMD) Systems and Multiple Instructions and Multiple Data (MIMD) Systems.

Fig. 2. A SIMD system consists of multiple simple processors with small local memory. These processors use explicit communication to transfer data to each other. All the different processors should be strongly synchronized. Because of the complexity and inflexibility, SIMD systems are not used for very advanced applications.

MIMD systems are more suitable to bioinformatics applications. In MIMD machines each process executes
completely independent of the other process asynchronously. MIMD systems are further classified on the bases of shared and distributed memory. A process running in the shared memory system can access any local or remote memory of the system whereas a process running in distributed memory cannot.

![Fig. 2 Summarized Parallel Computer Architectures.](image)

Shared memory systems have many advantages for bioinformatics applications. Design of parallel programs is simplified with a single address map. Different processes can also communicate without any time loss, because every CPU has direct access to memory. Whereas in distributed memory systems a time penalty is incurred for intercrosses communication because of the lack of a single address map for the memory.

Current trends in multiprocessor design try to utilize the positive factors of both the architectures. Each CPU has some local memory attached to it and hardware creates an illusion of common memory shared by the whole system. So the memory installed in any node may be accessed by any other node with very low less time penalty.

But as now very fast processors are available in the work stations, so microcomputers are connected with the help of Local Area Network. In this way virtual parallel computers are developed. These computers are also called Multi- computers which are constructed with the help of Cluster of Workstations (COWs). One more architecture of multi – computers is Beowulf – clusters which consist of very simple hardware components like ordinary PCs. In this architecture a public domain server controls the whole cluster.

**Parallelized Implementation Of Different Bioinformatics Algorithms**

**Cluster Implementation of Sequence Alignment Algorithms**

Smith waterman algorithm is used for local alignment between two sequences [7]. The algorithm is based on dynamic programming technique. If the two sequences of size $n$ are to be matched then algorithm takes time $O(n^2)$. As the value of $n$ increases the time required becomes significantly high. Thus the need of parallel implementation of Smith waterman algorithm arises [8]. In smith waterman algorithm is implemented on clusters. The results of this cluster implementation are shown in Table 1.

![Table 1 Parallel Implementation of Sequence Alignment Algorithms on Clusters](image)

In parallelization of the algorithm pipelining is used. In the score matrix, each row is computed sequentially and is blocked till the required cells in the above row are computed. When 32 processors are used with a sequence of 5000 characters long, the implementation showed an improvement up to 10.30 times. Smith waterman algorithm is also parallelized by its implementation on cell broadband engine [9]. In this implementation a static load balancing strategy is used. Under this strategy, work load at the beginning is divided equally among all the processors and processes. In the first step, algorithm reads the input dataset. In the next step the input sequences are processed by processing units to acquire the respective sequence parts in their local memories. For a sequence of 2048 characters long with this algorithm a speed up of 6.5 times is obtained. For multiple sequence matching multiple sequence alignment algorithms are used [10]. If there are $n$ sequences, $n(n-1)/2$ pair-wise alignments need to be calculated. As the number of sequences increase, number of pair wise alignments also increase and the complexity of the algorithm also.

Once the distance matrix is calculated, in the next phase of the algorithm phylogenetic tree is produced. And in the final phase of the algorithm, previously generated phylogenetic tree is used to determine the order of the alignment. Experiments were performed with a number of techniques and concluded that to distribute all the $n$ sequences to each processor was a better method. In this technique each of the $P$ processors performs exactly $n(n-1)/2P$ alignments. Although this method has very high communication cost, even then it showed maximum speed up. For $n = 500$ sequences, where each sequence had 200 characters this technique showed a speed up of 5.81 times [11]. The result of implementation on cluster is shown in Fig. 3.
Parallel multiple sequence alignment was also performed on the cell broadband engine [12] where the parallel portions of the code were executed on synergistic processing units whereas sequential code on power processing units. For n = 8 pair of sequences where each sequence had 2048 characters showed a speed up of 46.37x times which is shown in Fig. 4.

When this implementation is compared with a sequential implementation on a desktop with 3.2 GHz Pentium 4 Processor, a speed up of 6.5x is obtained. When this implementation was compared with best sequential algorithm with single SPU and a Pentium 4 Processor the speed ups were 4.5x and 3.5x respectively [11].

Cell Broadband Engine Implementation of FASTA for Multiple Sequence Alignment

Cell broadband engine based implementation for global alignment was performed on IBM Cell SDK 3.0 to obtain the results the implementation was executed on Sony Play Station 3 (SP3) and was compiled with optimized level -O3. The performance of this implementation was studied on different number of SPUs. The result of the implementation is shown in Fig. 5 by using up to 6 synergistic processing units (SPUs). When this implementation is compared with sequential implementation on a desktop with 3.2 GHz Pentium 4 Processor, a speed up of 6.5x is obtained. When this implementation was compared with best sequential algorithm with single SPU and a Pentium 4 Processor the speed ups were 4.5x and 3.5x respectively [11].

Once the alignment scores are calculated with the help of power processing units, scores, query and library sequences are delivered to synergistic processing unit to execute the smith waterman kernel. But this cell implementation is limited by the size of the sequence. A sequence of more than 2048 characters cannot be compared in this implementation because of the size of the synergistic processing unit local memory. This problem can be rectified with the help of pipeline approach. Once smith waterman is implemented on the cell, then it can be used in FASTA package. In FASTA each query sequence is compared with every sequence in the database. Hence balancing load between each pair of sequences is evaluated.

Protein Structure Prediction Algorithms

The most important application of protein structure prediction is drug design. In protein structure prediction tertiary structure of the proteins is predicted from its amino acid sequences. On the bases of physical properties many protein structures are possible. So it is very difficult to understand the stability of a structure.

Genetic algorithm is used to implement protein structure prediction on computational grid [13].
implementation of protein structure prediction is also done [14]. In this implementation sequences are shifted from database to synergistic processing units with this implementation a speed up between 3.2x and 3.6 x was achieved.

Conclusions
It is concluded that Parallel Computing is having very good impact on computational and data intensive applications. The processing time of bioinformatics algorithms can be improved by parallelization. The sections of the algorithms which take more time can be divided in to subprograms to execute concurrently.

References