SURVEY OF BIOLOGICAL HIGH PERFORMANCE COMPUTING: ALGORITHMS, IMPLEMENTATIONS AND OUTLOOK RESEARCH

Nasreddine Hireche, J.M. Pierre Langlois and Gabriela Nicolescu
Département de Génie Informatique, École Polytechnique de Montréal
{nasreddine.hireche, pierre.langlois, gabriela.nicolescu}@polymtl.ca

Abstract

During recent years there has been an explosive growth of biological data coming from genome projects, proteomics, protein structure determination, and the rapid expansion in digitization of patient biological data. Powerful computational techniques are required to understand and analyze biological information encoded by DNA sequences, which are frequently compared and searched for matching or near-matching patterns.

Comparison of DNA sequences and genes can be useful to investigate the common functionalities of the corresponding organisms and to get a better understanding of how specific genes or groups of genes are organized. This kind of similarity calculation is known as sequence alignment and its objective is to identify similarities between subsequences of strings. Gene sequence alignment is one such problem that serves as an initial step in many of the problems in bioinformatics.

Solving computational biology problems can be accelerated by algorithmic improvements or with the help of high-performance computing architectures. Such architectures include superscalar uniprocessors, parallel systems and dedicated hardware implementations of algorithms. FPGAs have emerged as high-performance computing accelerators, capable of implementing massively parallelized versions of computationally intensive algorithms. Their reprogrammability allows different algorithm-specific computing architectures to be implemented using the same hardware resource.

In this article we provide a state of the art review for this field of research. We identify specific algorithmic problems and how hardware architectures can be designed to solve them. We present systems recently reported, describe their main features, and provide a comparison between them. Finally, we offer some directions for future investigations.

Keywords*: DNA sequences alignment, dynamic programming algorithms, accelerators, FPGAs.*

1. Introduction

With the advent of the genome era, bioinformatics plays important roles in biological and medical research. Computational biology and the organization of biological data related to genomes aims to apply this information to the determination of disease origins, pharmacology, and other commercial applications. Bioinformatics focuses on the development of practical tools for data management and analysis.

Sequences alignment is the procedure of comparing two DNA or protein sequences by searching for a series of individual characters or character patterns that are in the same order in the sequences. This problem is often addressed using dynamic programming (DP) algorithms [1]. DP algorithms are guaranteed, in a mathematical sense, to provide the optimal (highest-scoring) alignment for a given set of user-defined variables, including choice of scoring matrix and gap penalties [1].

Popular algorithms for this operation are heuristic approaches such as BLAST [2] and FASTA [3], which are fast but have low sensitivity. The Smith-Waterman (SW) algorithm [4] is a more computationally expensive algorithm but it achieves higher sensitivity. Sencel Bioinformatics [5] compared the sensitivity and selectivity of various searching methods. The sensitivity was measured by the coverage, which is the fraction of correctly identified homologues. These experiments show that for coverage around 0.18, the errors per query of BLAST and FASTA are about two times that of the SW algorithm [6].

There are two types of pair-wise sequences alignments, global and local [1]. In local alignment (LA), stretches of sequences with the highest density of matches are aligned, thus generating one or more islands of matches or sub alignments. LAs are more suitable for sequences that are similar along some of their lengths but dissimilar in others. The SW algorithm produces the LAs between substrings of two sequences $N[1..n]$ and $M[1..m]$, of lengths m and n, respectively, using a DP approach [4]. The strategy consists of building a solution gradually using simple recurrences by taking the maximum score of the similarity of $N[1..n\text{-}1]$ and $M[1..m\text{-}1]$ plus one of three following values: substitution *(sub)*, deletion *(del) N[n]*, and insertion *(ins) M[m]*.

To solve the problem with this recurrence, the algorithm builds an $(n+1) \times (m+1)$ score matrix S, where each $S[i, j]$ represents the similarity between sequences $N[1..i]$ and $M[1..j]$. The alignment stops at the ends of regions of strong similarity, and a much higher priority is given to finding these local regions than to extending the alignment to include more neighboring nucleotides or amino acid pairs. In this algorithm, only positive values are kept. Any negative values are converted to zero, which has the effect of terminating any alignment up to that point. The first row and the first column

1-4244-0038-4 2006
IEEE CCECE/CCGEI, Ottawa, May 2006

represent alignments of one sequence with spaces. $S[0, 0]$ represents the alignment of two empty sequences, and is set to zero. All other entries are computed with the following formula [7]:

$$S[i, j] = \max \begin{cases} S[i-1, j-1] + sub(N[i], M[j]) \\ S[i, j] + del(N[i]) \\ S[i-1, j] + ins(M[j]) \\ 0 \end{cases}$$

A trace-back matrix is used to keep track of the moves in the scoring matrix. Alignments are produced by starting at the highest-scoring position in the scoring matrix and following a trace path from those positions up to a box that scores zero.

For two sequences of lengths m and n, there are $(n+1) \times (m+1)$ positions to be computed. The algorithm thus has a time complexity of $O(n^2)$. It has also quadratic space complexity because the entire matrix must be kept in memory.

2. Bioinformatics algorithms implementation considerations

One of the main goals of designers of biological high performance computing is to develop a platform for efficient biological sequence analysis and especially for the SW algorithm. High-performance sequence analysis often relies on straightforward parallelization of the $O(n^2)$ DP algorithm. A common mapping is to assign one processing element (PE) to each character of the query string, and then to shift the database through the linear chain of PEs. A set of PEs can form a reprogrammable and flexible accelerator when these compromises are required. The implementation of the SW algorithm always presents a system design challenge to improve biological high performance computing using different analysis approaches.

Moore's Law observes that the number of transistors that can be integrated doubles every 18 months [8]. However, the amount of genomic data at GenBank, an annotated collection of all publicly available DNA sequences, is doubling every six months [9]. The growing gap between the amount of information to be analyzed and integration density implies that different implementation strategies must be considered. This requires a massive design effort to choose the best compromise between three criteria: flexibility, programmability and computational density.

Flexibility implies the possibility to reuse hardware for different applications. Programmability is a system-level attribute pertaining to its ability to support reconfiguration as required. Computational density is a hardware/software attribute referring to the amount of computation per area, volume, or power. It can correspond inversely to the price per unit of performance.

Considering these three criteria, four common types of processing systems can be used for sequence alignment processing [10]:

- *General-purpose supercomputers* are the most flexible means of fast sequence analysis, but they suffer from very high cost.
- *Single-purpose processors* can achieve the highest performance for a single algorithm for prices in the low tens of thousands of dollars.
- *Programmable co-processors* strive for the algorithmic flexibility of reconfigurable systems and the speed and density of single-purpose systems. Cost and ease of programming generally fall between the other two types. ASIC-based accelerators have the disadvantage of low production volume, and therefore the design costs cannot be spread over very many units. Examples of this type of accelerator are Kestrel [11], and GeneMatcher 2 [12].
- *Reconfigurable Hardware* includes systems based on FPGAs. They tend to have a much lower processing element density than single-purpose processors, but they can be faster than supercomputers. FPGA and related compiler technology is improving and FPGA-based accelerators are much less expensive to design. An example of a recent FPGA-based accelerator is the proprietary DeCypher [13], on which little detailed design information has been published.

3. Hardware architectures review and comparisons

In this section we present five hardware architectures designed specifically for sequence analysis.

The first one is Splash, described by M. Gokhale et al. [14]. A programmable linear logic array, it bridges the gap between the traditional fixed-function VLSI systolic array and the more versatile programmable array. Splash received a 1989 Gordon Bell Prize honorable mention for timings on a problem that compared a new DNA sequence against a library of sequences to find the closest match. The systolic array consists of many stages connected in a one-dimensional array. Each stage has three components: a Xilinx FPGA, local memory, and a floating-point chip. In the Splash implementation, a processing element is composed of two modules: a character comparator and a finite state machine.

The second system that we present is from B. Schmidt et al. [15]. This system targets high performance protein database scanning on novel massively parallel architectures to gain supercomputer power at low cost. The first architecture is built around a PC cluster linked by a high-speed network and fine-grained parallel Systola 1024 processor boards connected to each node. The second architecture is the Fuzion 150, a parallel computer with a linear single-instruction, multiple-data stream (SIMD) array of 1536 processing elements on a single chip. The authors presented the design of a database scanning application based on the SW algorithm in order to derive efficient mappings onto these architectures. The implementations lead to significant runtime savings for large-scale database scanning [15].

The third system is the UCSC Kestrel parallel processor [11]. It is a single-board coprocessor with a 512-element linear

array of 8-bit SIMD processing elements. The system was designed and built at the University of California at Santa Cruz in 1993-1995, where bioinformatics applications motivated development of a sequence analysis engine that could efficiently analyze genomic databases. The architectural decisions surrounding Kestrel led to a particularly high density of computation that enables the single-board system with 9 million transistors of custom VLSI, an FPGA, and various memory chips, to outperform a current workstation 10 years later [11].

The fourth system we discuss is the S. Smith and J. Frenzel single-chip shared-memory multiprocessor architecture [10]. The principle in this architecture is to put many small and simple processors on a large integrated circuit. A system bus allows data to pass between the processors and a shared memory. For the SW algorithm application, this shared memory system may only be a cache for a much larger memory located off the integrated circuit. The database items are only read once and each result is only written once, so traffic between off-chip memory and on-chip shared memory cache is low.

The fifth and last system in this state of the art review is the Nallatech-Configurable Multi-FPGA System [7]. It is a three-FPGA network board comprising a BenNUEY motherboard with a Virtex-II XC2V3000-4 FPGA and an attached BenBLUE-II daughterboard with two Virtex-II XC2V6000-4 FPGAs. In this system the SW algorithm is implemented providing a computational speed improvement of more than two orders of magnitude (200X) [7]. The query sequences are hard-coded into the FPGAs while the individual database sequence files are loaded in the main memory. The database sequences are first written across the DIMEtalk network into the 16K read-FIFOs of the individual FPGAs. A single XC2V6000 device is capable of as many as 1.26 trillion cell updates per second. At the end of the processing, the final edit distance is written into a write-FIFO.

The performance of sequences comparison systems is commonly measured on millions of processing element or cell updates per second (MCUPS) [16]. Table 1 gives a summary of the systems mentioned in this paper, regarding their performance, design approaches, and their different technological classification [10][15].

Table 1. Performance of architectures applying the SW algorithm

Platform	Year	Technology	PEs	MCUPS
Splash II	1993	1000 nm FPGA	1536	3000?
Kestrel	1997	500 nm ASIC	512	400
Fusion 150	2000	250 nm ASIC	1536	2500
DeCypher	2001	180 nm FPGA	N/A	7500
GeneMatcher2	2001	130 nm ASIC	2872	16000

With the exception of the first architecture, the comparison of systems characteristics given in Table 1 shows a progressive increase in MCUPS matching improvements in process resolution for ASICs and FPGAs. Overall system performance is very close between these two technologies, but the great inconvenient of ASICs is their very high design costs, when FPGA systems are cheap and reconfigurable.

4. Future directions

The use of accelerators with an aim of increasing the execution speed of specific functionalities is not new, but the specific challenges of high performance calculation in bioinformatics require new solutions. The reduction of power consumption is also an important consideration. New technological evolutions highlight the importance of discharging general-purpose processors from tasks with great computational density, such as those related to biological databases.

General-purpose processors cannot respond efficiently to these challenges by themselves. In recent years, the increase in clock frequency of general-purpose processors has slowed down [17], but the transistor integration density has not. This creates two reasons explaining in part the trend towards the use of processing accelerators. The first is the fact that the accelerator can be coupled and integrated on chips along with a CPU in order to have the same technology advances such as low power consumption and best space deployment. The second and main advantage is that the integration level progression implies a great density of processing elements and a very high effective combined frequency of operation.

For instance, for CMOS technology, the International Technology Roadmap for Semiconductors identifies process resolution of 130 nm, 70 nm, and 22 nm for the years 2003, 2006 and 2016, respectively [8]. The corresponding number of processing elements that could be implemented for the application of S. Smith [10] would be 227, 782 and 5,778. Taking into account the corresponding increases in operating frequency, the equivalent combined processing frequencies are 57, 644 and 24300 GHZ. These high processing frequencies can be divided by the clock cycles per database character rate to measure the performance of systems by millions of dynamic programming cell updates per second (MCUPS).

The integration of such a large number of processing elements for the implementation of different algorithms requires sophisticated design techniques and system management.

The potential of FPGAs as reconfigurable computing accelerators arises from the fine-grained parallelism required for each incremental compute performance increase [18]. With that extra efficiency should come higher computational throughput. FPGA performance is increasing by 4× every two years, and, for operations that use architectural improvements, performance is increasing at a rate of 5X every two years [19]. Thus, FPGAs have the potential to offer a global performance increase of 20× every 2 years. Human genetic database contents are doubling every six months [9], a 16× growth over two years, so it appears that FPGA technology improvements can respond to the human genetic data explosion challenge.

For reconfigurable computing, a considerable design methodology definition effort remains to be done, because the

middle design levels are missing between the logic structures and Medium Scale Integration levels and the applications layer [18]. There is not yet the equivalent of BIOS for an FPGA-based reconfigurable computer that would isolate the OS from the particulars of each specific hardware configuration. This would be useful for supporting the development of any future OS-like layer that might run on top of an FPGA-based reconfigurable processor [18].

5. Conclusion

In this paper we presented bioinformatics applications trends, the associated high performance computational challenges, and algorithms that require efficient computing systems. We provided a classification of the existing analysis approaches, a state of the art of existing systems in this domain, and we explained accelerator trends. We observed that the preferred way to improve system performance is by increasing the global combined frequency. Using more processing elements by accelerator most effectively does this. This depends on the integration level and the optimal technical design.

With a comparison study, we observed that FPGA performances are similar those of ASICs, which have a very high cost. In addition to their affordability, FPGAs offer performance that can respond to the explosive growth in biological data and the associated future challenges. However, a significant design effort remains for system level development.

References

[1] D.W. Mount, *BIOINFORMATICS: SEQUENCE AND GENOME ANALYSIS,* Second Edition. Cold Spring Harbor Laboratory Press, New York, 2004.

[2] National Center for Biotechnology Information. BLAST home page. www.ncbi.nlm.nih.gov/blast, Jan. 2006.

[3] European Bioinformatics Institute Home Page, www.ebi.ac.uk/fasta33/, "FASTA Searching program", Jan. 2006.

[4] N.C. Jones and P.A. Pevzner, *AN INTRODUCTION TO BIOINFORMATCS ALGORITHMS.* MIT Press, Cambridge, Massachusetts London, England, 2004.

[5] Sencel's search software, www.sencel.com, Jan. 2006.

[6] C.W. Yu, K.H. Kwong, K.H. Lee and P.H.W. Leong, "A Smith-Waterman systolic cell," *Proceedings of the Tenth International Workshop on Field Programmable Logic and Applications (FPL'03),* pp. 375-384, 2003.

[7] K. Regester, J-H. Byun, A. Mukherjee, and A. Ravindran, "Implementing bioinformatics algorithms on Nallatech-configurable multi-FPGA systems," *Xcell Journal Online,* March 2005.

[8] International Technology Roadmap for Semiconductors, *PROCESS INTEGRATION, DEVICES, AND STRUCTURES,* 2003 Edition.

[9] GenBank growth statistics, www.ncbi.nlm.nih.gov, Feb. 2005.

[10] S.F. Smith and J.F. Frenzel, "Bioinformatics searches using a single-chip shared-memory multiprocessor," *International Conference on Parallel and Distributed Processing Techniques and Applications,* 2003.

[11] A.D. Blas et al., "The UCSC Kestrel parallel processor," *IEEE Transaction on Parallel and Distributed Systems,* Vol. 16, N. 1, Jan. 2005.

[12] Paracel, Inc., www.paracel.com, Oct. 2005.

[13] TimeLogic Corp., www.timelogic.com, Jan. 2006.

[14] M. Gokhale, W. Holmes, A. Kopser, S. Lucas, R. Minnich, D. Sweely, and D. Lopresti, "Building and using a highly parallel programmable logic array," Computer, vol. 24, pp. 81-89, Jan. 1991.

[15] B. Schmidt, H. Schroder, and M. Schimmler, "Massively parallel solutions for molecular sequence analysis," *Proc. Int'l Parallel and Distributed Processing Symp*., pp. 186-192, Apr. 2002.

[16] R. Hughey, "Parallel Sequence Comparison and Alignment," *IEEE Computer Society,* Jul. 1995.

[17] C. Johnson, J. Welser, "Future processors: flexible and modular," *3rd IEEE International Conference on Hardware Software Codesign,* Sept. 2005.

[18] K. Morris, "Saving Supercomputing with FPGAs," *FPGA and Structured ASIC Journal,* Nov. 2005.

[19] A. Schnore, M. Devlin, "SC05 OpenFPGA BOF", www.Openfpga.com, Nov. 2005.