BIOINFORMATICS APPLICATIONS NOTE Vol. 21 no. 16 2005, pages 3431–3432
doi:10.1093/bioinformatics/bti508

Sequence analysis

Using reconfigurable hardware to accelerate multiple sequence alignment with ClustalW

Tim Oliver[1,*], Bertil Schmidt[1], Darran Nathan[2], Ralf Clemens[2] and Douglas Maskell[1]

[1]School of Computer Engineering, Nanyang Technological University, Singapore and [2]Project Proteus, School of Engineering, Ngee Ann Polytechnic, Singapore

Received on March 21, 2005; revised on April 29, 2005; accepted on May 18, 2005
Advance Access publication May 26, 2005

ABSTRACT

Summary: Aligning hundreds of sequences using progressive alignment tools such as ClustalW requires several hours on state-of-the-art workstations. We present a new approach to compute multiple sequence alignments in far shorter time using reconfigurable hardware. This results in an implementation of ClustalW with significant runtime savings on a standard off-the-shelf FPGA.

Availability: An online server for ClustalW running on a Pentium IV 3 GHz with a Xilinx XC2V6000 FPGA PCI-board is available at http://beta.projectproteus.org. The PE hardware design in Verilog HDL is available on request from the first author.

Contact: tim.oliver@pmail.ntu.edu.sg

INTRODUCTION

Progressive alignment is a widely used heuristic method to compute multiple sequence alignments (MSAs). Progressive alignment basically consists of three steps. First, a distance value between each pair of input sequences is computed (pairalign). Second, a phylogenetic tree is calculated based on the distance matrix (guided tree). Finally, pairwise alignment of various profiles is carried out following the branching order in the phylogenetic tree to form the final MSA (malign). A popular tool based on this method is ClustalW (Thompson *et al*., 1994). Unfortunately, progressive alignment programs suffer from high-computational complexity, for instance the alignment of a few hundred protein sequences using ClustalW requires several hours on a state-of-the-art workstation. Owing to the rapid growth of sequence databases, biologists would like to compute MSAs of an increasing number of sequences in reasonable time.

Consequently, several parallel implementations of ClustalW have been developed to speed up this time-consuming task. The solutions recently presented by Li (2003) and by Ebedes and Datta (2004) use message-passing on a PC cluster. Parallel ClustalW implementations have also been designed for more expensive shared memory machines (Mikhailov *et al*., 2001; Duzlevski, 2002, http://bioinfo.pbi.nrc.ca/clustalw-smp/).

The parallelization strategy taken in this paper is based on field programmable gate arrays (FPGAs). FPGAs provide a flexible platform for fine-grained parallel computing based on reconfigurable hardware. Since there is a large overall FPGA market, this approach has a relatively small price/unit and it also facilitates regular upgrading to FPGAs based on state-of-the-art technology. The usefulness of this technology for fast sequence analysis has already been recognized (Marongiu *et al*., 2003). Previous work has mainly focused on high speed homology searching (see e.g. Huang, 1993; Yamaguchi *et al*., 2002).

*To whom correspondence should be addressed.

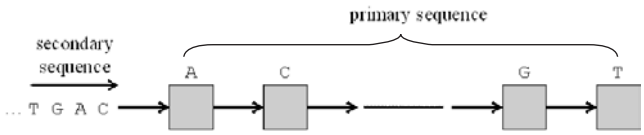

Fig. 1. Pairwise distance computation on a linear PE array: a primary sequence is loaded into the processor array and a secondary sequence flows from left to right through the array.

METHODS

Profiling the three stages of ClustalW for different numbers of input sequences reveals that >90% of the overall runtime is spent on the first stage (pairalign). Hence, we have decided to accelerate only this stage using reconfigurable hardware.

Our mapping strategy uses a fine-grained parallelization of the pairwise distance computation. ClustalW derives the distance score of a pair of sequences from the number of matching characters in their optimal local alignment. Sequentially, this score can be computed by locating the maximum value in the Smith–Waterman dynamic programming (DP) matrix and then performing a trace back procedure to produce the corresponding alignment. However, the trace back operation is far too control intensive for efficient FPGA acceleration. We have therefore extended the Smith–Waterman recurrence relation to include the counting of exact matches during computation of the DP matrix.

The DP calculation exhibits a high regularity and can be efficiently mapped onto a linear array of small processing elements (PEs). One PE is assigned to each character of a (primary) sequence. A (secondary) sequence is then systolically shifted through the linear chain of PEs (Fig. 1). Each PE is capable of computing one value for the DP matrix within each clock cycle. Amino acid sequences are usually longer than the number of PEs that can fit on a reconfigurable hardware chip. Therefore, we have partitioned the computation on a fixed-sized processor array. The partitioning is implemented by means of a loop FIFO memory that connects the right PE with the left PE.

RESULTS

We have described the hardware design in Verilog. Using a Xilinx Virtex II XC2V6000, we are able to accommodate 92 PEs at a maximum allowable clock speed of 34 MHz. Our implementation achieves a sustained performance (including all data transfer) of ∼1 GCUPS (billion cell updates per second in the DP matrix), a

Table 1. Comparison of runtimes (in seconds) and speedups of ClustalW running on a single Pentium IV 3 GHz with our FPGA-accelerated version running on a Pentium IV 3 GHz with a Xilinx XC2V6000 FPGA for a different number of input sequences (globins taken from Swiss-Prot)

Number of protein sequences (average length)	200 (412)	400 (468)	600 (462)	800 (454)	1000 (446)
ClustalW (Pentium IV, 3 GHz)					
Overall	194.9	891.9	1818.1	3157.6	4711.6
Pairalign	183.8 (94.4%)	833.1 (93.4%)	1697.0 (93.3%)	2966.6 (94%)	4409.6 (93.6%)
Guided tree	0.07 (0.03%)	0.8 (0.09%)	4.1 (0.2%)	8.0 (0.2%)	16.0 (0.3%)
Malign	11.0 (5.6%)	58.0 (6.5%)	117.0 (6.4%)	183.0 (5.8%)	286.0 (6.1%)
FPGA-accelerated ClustalW (Pentium IV, 3 GHz with Xilinx XC2V6000)					
Overall	14.7	76.9	159.3	256.0	399.5
Pairalign	3.6 (24.6%)	18.1 (23.5%)	38.2 (24%)	65.0 (25.4%)	97.5 (24.4%)
Guided Tree	0.07 (0.5%)	0.8 (1%)	4.1 (2.6%)	8.0 (3.1%)	16.0 (4%)
Malign	11.0 (74.9%)	58.0 (75.4%)	117.0 (73.4%)	183.0 (71.5%)	286.0 (71.6%)
Speedup					
Overall	13.3	11.6	11.4	12.3	11.8
Pairalign	50.9	46.0	44.4	45.6	45.2

We have used the ClustalW code from Li (http://web.bii.a-star.edu.sg/ kuobin/clustalw-mpi/index.html) for our evaluation.

measure commonly used to compare parallelized Smith–Waterman implementations. A set of performance evaluation tests have been conducted using a different number of globin sequences to evaluate the processing time of the FPGA-accelerated implementation versus that of the sequential ClustalW code. The PCI based ADP-WRC-II board from Alpha-Data with a Xilinx XC2V6000 FPGA has been used in the tests. The ClustalW application is benchmarked on an Intel Pentium IV 3 GHz processor with 1 GB of RAM. The results for this are shown in Table 1. We have also set-up a web server for ClustalW running on this hardware configuration, which is available online at http://beta.projectproteus.org.

The FPGA-accelerated pairwise alignment stage achieves speed-ups between 45 and 50. At least the same number of PCs connected by a fast switch is required to achieve a similar speedup using the ClustalW-MPI code from Li (2003). A comparison of these two parallelization approaches shows that reconfigurable hardware acceleration is superior in terms of price/performance. Our solution is also easily scalable to next-generation FPGAs such as the Virtex-4 family by simply increasing the number of PEs in our design.

Table 1 also shows that the progressive alignment stage of ClustalW (malign) dominates the runtime of the FPGA-accelerated ClustalW. This stage comprises computation of several profile–profile alignments based on DP. It will be interesting to investigate how this can be efficiently mapped onto reconfigurable hardware. The first stage of several other MSA tools such as T-Coffee (Notredame *et al*., 2000) and MUSCLE (Edgar, 2004) is also based on the computation of pairwise distances. Hence, these tools will see a similar speedup from the accelerator presented in this paper.

Conflict of Interest: none declared.

REFERENCES

Duzlevski,O. (2002) SMP version of ClustalW 1.82, unpublished.

Ebedes,J. and Datta,A. (2004) Multiple sequence alignment in parallel on a workstation cluster. *Bioinformatics*, **20**, 1193–1195.

Edgar,R.C. (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acid Res.*, **32**, 1792–1797.

Huang,D.T. (1993) Searching genetic databases on Splash -2. In *Proceedings of the IEEE Workshop on Custom Computing Machines*. IEEE Computer Society Press, pp. 185–191.

Li,K.-B. (2003) ClustalW-MPI: ClustalW analysis using distributed and parallel computing. *Bioinformatics*, **19**, 1585–1586.

Marongiu,A. *et al*. (2003) Designing hardware for protein sequence analysis. *Bioinformatics*, **19**, 1739–1740.

Mikhailov,D., Cofer,H. and Gomperts,R. (2001) *Performance Optimization of ClustalW: Parallel ClustalW, HT Clustal, and MULTICLUSTAL*. White papers, Silicon Graphics, CA.

Notredame,C. *et al*. (2000) T-Coffee: a novel method for fast and accurate multiple sequence alignment. *J. Mol. Bio.*, **302**, 205–217.

Thompson,J.D. *et al*. (1994) ClustalW: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.*, **22**, 4673–4680.

Yamaguchi,Y. *et al*. (2002) High speed homology search with FPGAs. *Pac. Symp. Biocomput.*, **2002**, 271–282.