

Accelerating the Kernels of BLAST with An Efficient PIM (Processor-In-Memory) Architecture

Jung-Yup Kang, Sandeep Gupta
 Department of Electrical Engineering
 University of Southern California
 Los Angeles, CA 90089
 jungyup@usc.edu

Jean-Luc Gaudiot
 Department of Electrical Engineering
 and Computer Science
 University of California at Irvine
 Irvine, CA 92709
 gaudiot@uci.edu

Abstract—BLAST is a widely used tool to search for similarities in protein and DNA sequences. However, the kernels of BLAST are not efficiently supported by general-purpose processors because of the special computational requirements of the kernels.

The kernels involve large amounts of computations which contain a high degree of potential parallelism that general-purpose processors can only exploit to a very limited extent. The kernels handle operands that are small (one byte) and not efficiently manipulated by general-purpose processors. The kernels entail only simple operations whereas current general-purpose processors expend significant proportion of their chip area to support complex operations, such as floating-point operations. The kernels perform a large amount of memory accesses, which translates into severe penalties.

In this paper, we propose an efficient PIM (Processor-In-Memory) architecture to effectively execute the kernels of BLAST. We propose not only to reduce the memory latencies and increase the memory bandwidth but also to execute the operations inside the memory where the data are located. We also propose to execute the operations in parallel by dividing the memory into small segments and by having each of these segments executes operations concurrently. Our simulation results show that our computing paradigm provides a $242\times$ performance improvement for the executions of the kernels and a $12\times$ performance improvement for the overall execution of BLAST.

Index Terms—BLAST, Processor-In-Memory (PIM) Architecture, Sequence Alignment.

SUMMARY

BLAST (Basic Local Alignment Search Tool) [1], [2], [10] is a molecular biology application where a database of protein (or DNA) molecule sequences is compared against a query sequence representing a reference protein molecule so as to uncover similarities. It is a highly time-consuming application simply because it deals with a large database of sequences and hence must perform a large number of memory accesses and computations [2], [9].

In the past, a variety of high performance machines such as NOW (Network Of Workstations) or massively parallel computers [6], [7], [8] have been used to speed up execution of BLAST. In such systems, each node, which is a general-purpose processor (GPP), is responsible for its share of the database. However, GPPs are not efficient in executing BLAST because they are susceptible to the memory wall problem [3], [4], [5], [11], [12] due to the large number of memory accesses in BLAST. At the same time, their limited concurrency prohibits them from exploiting the tremendous amount of parallelism inherent in BLAST. Executing in GPPs is also wasteful because the basic operand size of the operations of BLAST (one byte), does not fully utilize the 32- or 64- bit functional units of GPPs. Moreover, since the majority of its operations are simple comparisons, accumulations, and subtractions, the complex functional units of GPPs, such as multipliers, dividers, and floating-point units remain unused during the execution of BLAST.

Therefore, we propose a hardware/software co-design paradigm which uses a PIM (Processor-In-

Memory) module to efficiently execute the kernels of BLAST. Our PIM module reduces the memory access penalty caused by the large number of memory accesses by relocating the data-intensive operations to where data are located instead of transferring data to execution units. We exploit the inherent parallelism of BLAST by segmenting the PIM module into many sub-modules and executing the kernels in each of these sub-modules in parallel. Therefore, we can execute the most time-consuming and the most memory bandwidth-intensive operations more efficiently. We also introduce a BLAST kernel specific data communication mechanism to overcome the obstacles of parallel execution.

Our simulation results show that by using our computing paradigm, there is a $242\times$ performance improvement for the execution of the kernels and a $12\times$ performance improvement for the overall execution of BLAST. Future work includes integrating reconfigurability in our computing paradigm to cover different kinds of BLAST programs. Our paradigm can be conveniently modified and similarly applied to support other BLAST programs.

ACKNOWLEDGMENT

This paper is based upon work supported in part by NSF grants CCR-0234444 and INT-0223647. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

- [1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic Local Alignment Search Tool. *Journal of Molecular Biology*, 215:403–410, 1990.
- [2] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: A New Generation of Protein Database Search Programs. *Nucleic Acids Res.*, 25:3389–3402, 1997.
- [3] D. Burger, J. R. Goodman, and A. Kägi. The Declining Effectiveness of Dynamic Caching for General-Purpose Microprocessors. Technical Report UWMADISONCS CS-TR-95-1261, University of Wisconsin-Madison, January 1995.
- [4] D. Burger, J. R. Goodman, and A. Kägi. Memory Bandwidth Limitations of Future Microprocessors. In *Proceedings of the 23rd International Symposium on Computer Architecture*, pages 78–89, 1996.
- [5] D. Burger, J. R. Goodman, and A. Kägi. Limited Bandwidth to Affect Processor Design. *IEEE Micro*, 17(6):55–62, 1997.
- [6] E. Glemet. LASSAP: A Large Scale Sequence comparison Package. *Bioinformatics*, 32(2):137–143, 1997.
- [7] D. Lavenier. SAMBA : Systolic Accelerator for Molecular Biological Applications. Technical Report RR-2845, IriSa, 1996.
- [8] D. Lavenier and J.-L. Pacherie. Parallel Processing for Scanning Genomic Data-Bases. In *Parallel Computing: Fundamentals, Applications and New Directions, Proceedings of the Conference ParCo'97, 19-22 September 1997, Bonn, Germany*, volume 12, pages 81–88, 1997.
- [9] National Center for Biotechnology Information. NCBI News, Spring 2003. *NCBI News*, 25:8, 2003.
- [10] NCBI. National Center for Biotechnology Information, <http://www.ncbi.nlm.nih.gov/>.
- [11] D. Patterson, T. Anderson, N. Cardwell, R. Fromm, K. Keeton, C. Kozyrakis, R. Thomas, and K. Yelick. A Case for Intelligent RAM. *IEEE Micro*, 17(2):34–44, 1997.
- [12] D. Patterson, N. Cardwell, T. Anderson, R. Fromm, K. Keeton, C. E. Kozyrakis, R. Thomas, and K. Yelick. Intelligent RAM (IRAM): Chips That Remember and Compute. In *ICCD '97 International Conference on Computer Design*, pages 224–225, 1997.