CONF 950484--2

Sequence Comparison on a Cluster of
Workstations Using the PVM System*

X. Guan, R. J. Mural†, E. C. Uberbacher

Informatics Group
Intelligent Systems Section
Engineering Physics and Mathematics Division and †Biology Division
Oak Ridge National Laboratory
P.O. Box 2008
Oak Ridge, Tennessee 37831-6364

"The submitted manuscript has been
authored by a contractor of the U.S.
Government under contract No.
DE-AC05-84OR21400. Accordingly, the
U.S. Government retains a nonexclusive,
royalty-free license to publish or
reproduce the published form of this
contribution, or allow others to do so,
for U.S. Government purposes."

Published Presentation: Submitted to the 9th International Parallel Processing
Symposium (IPPS '95), Santa Barbara, CA, April 25-28, 1995.

MASTER

*Research was supported by the Office of Health and Environmental Research, U.S.
Department of Energy under contract No. DE-AC05-84OR21400 with Martin Marietta
Energy Systems, Inc.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Q W Q

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Sequence Comparison on a Cluster of Workstations Using the PVM System*

X. Guan, R. J. Mural¹, E. C. Uberbacher

Engineering Physics and Mathematics Division
and ¹Biology Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6364

Abstract

We have implemented a distributed sequence comparison algorithm on a cluster of workstations using the PVM paradigm. This implementation has achieved similar performance to the intel iPSC/860 Hypercube, a massively parallel computer. The distributed sequence comparison algorithm serves as a search tool for two Internet servers GRAIL and GENQUEST. This paper describes the implementation and the performance of the algorithm.

* Research was supported by the Office of Health and Environmental Research, U.S. Department of Energy under Contract No. DE-AC05-84OR21400 with Martin Marietta Energy System, Inc.

Introduction

The most important information about a newly discovered DNA sequence is obtained by comparing it to the sequences in DNA and protein databases. The results of the sequence comparison can direct further studies on the sequence [8]. The optimal algorithms for sequence comparison are variations of the dynamic programming algorithm by Smith and Waterman [9]. As its complexity is $O(mn)$, where m and n are the lengths of the two sequences compared, the dynamic programming algorithm is also very slow. Furthermore, as the Human Genome Project develops, large amounts of DNA and protein sequences are being produced. The size of the current DNA sequence database is about 180 million bases. Searching a small DNA sequence (500 bases) against the entire database using the full dynamic programming algorithm takes more than two days on a SUN workstation (SPARC 20).

There are some heuristic methods such as BLAST [1] and FastA [7] that trade accuracy and sensitivity for speed, but to detect subtle and distant relationships, where many insertions and deletions are involved, it is usually the dynamic programming algorithm that is relied upon.

In order to speed up the dynamic programming algorithm, efficient implementations on various computers have been investigated. The most efficient methods employ massively parallel computers with hundred or thousands of processors, which are not readily available to meet the researcher's daily computational needs.

An alternative way of solving the problem is to cluster a group of already available workstations. Software systems such as PVM (Parallel Virtual Machine) [2] make it relatively easy to integrate workstations into a computing engine capable of solving large computationally-intensive problems. The cluster system is robust in the sense that workstations can be easily added or deleted from the PVM system, and if one workstation goes down, its share of the work can be easily transferred to other workstations.

The Dynamic Programming Algorithm

Sequence comparison can be used to find the conserved regions (subsequences or substrings) between two sequences, or to study the overall relationships between sequences by aligning the two sequences character by character. The former is called local sequence alignment, the latter global sequence alignment. The dynamic programming algorithm used most often for local alignment is the one proposed by Smith and Waterman [9]. A version of the Smith-Waterman algorithm modified by Gotoh [6] reduced the complexity of the original algorithm from $O(m^2n)$ to $O(mn)$: given two sequences $A = a_1a_2...a_m$ and $B = b_1b_2...b_n$, a cost, w , for k insertions or deletions is defined as $w(k) = -(u * k + v)$, where $u \geq 0$ and $v \geq 0$.

$$D(i, j) = \max \begin{cases} D(i-1, j-1) + c(b_i, a_j), \\ P(i, j), \\ Q(i, j). \end{cases} \quad 0 < i \leq m, 0 < j \leq n$$

where

$$P(i, j) = \max \begin{cases} D(i-1, j) + w(1), \\ P(i-1, j) + u. \end{cases}$$

$$Q(i, j) = \max \begin{cases} D(i, j-1) + w(1), \\ Q(i, j-1) + u. \end{cases}$$

$$P(0, k) = Q(k, 0) = D(k, 0) = D(0, k) = 0, \forall k > 0$$

and

$$c(b_i, a_j) = \begin{cases} \leq 0 & \text{if } b_i \neq a_j \\ \geq 0 & \text{if } b_i = a_j \end{cases}$$

To find the best local alignment, the maximum score in the matrix D is found first, then a traceback procedure is performed to find the actual alignment. If only the maximum score and the two ends of the best local alignment are needed, the algorithm can be implemented in linear space.

PVM

PVM is a software package developed at Oak Ridge National Laboratory [2]. It allows distributed computing on networks of computers, including workstations and supercomputers. It utilizes message passing mechanism for communication between participating workstations and computers on the network. As is common in a work place, workstations are often not fully utilized, or have various loads, and often they sit idle during off hours. The computing resources on these workstations can be used for computational problems such as sequence comparison.

Figure 1: A cluster of workstations connected by PVM system.

We have been using an intel iPSC/860 massively parallel computer as our computing engine for sequence comparison for two Internet servers GRAIL [4] and GENQUEST [5]. The two servers have a large user community with users around the world. Since the iPSC/860 is still a scarce resource used by many other users, a user's request may not always be processed in a timely manner, and a PVM-based implementation using existing SUN and DEC ALPHA workstations is desirable as a back-up. (see Figure 1).

Since some of these machines are used for computing problems, and others are used for managing files or editing, the loads on the workstations are not even. This presents a problem known as load balancing which will be addressed in the next section.

Problem Decomposition and Implementation

1. Data parallel approach:

In a sequence comparison problem, a query sequence is compared to every sequence in a database, so a data parallel approach is a natural choice. The target database is partitioned among the workstations according to parameters such as speed, load, communication, etc. Each processor proceeds to search its portion of the database. When the search finishes, a global sort is performed to find the global top matching sequences from the local search results. This is the so-called static decomposition approach, and has been used successfully on an iPSC/860 [3]. But on a cluster of workstations, loads are more dynamic, and the static decomposition approach is not flexible enough to adapt to changes in the computing environment. For example, if a user logs into a workstation, that workstation's load is increased, thus slowing down the process of the PVM task assigned to it, which in turn drags down the overall system performance. The static decomposition approach works well if the computing tasks are running while the cluster is idle (such as during the off hours).

2. Manager-worker approach:

Another approach is to consider the cluster as a pool of computing resources. One resource serves as a task manager (it could also be a worker), and the rest as workers. The task manager divides the whole problem into smaller portions (tasks), and the workers compete for the tasks. Because speeds and loads of these workstations are not even, fast workstations process more tasks, and slow ones get fewer tasks. Furthermore, the system adapts to the changing load situations. If a workstation is slowed down due to additional user's activities, that workstation will request fewer tasks from the manager, and as a result, other workstations take over the tasks which would have been assigned to that workstation under normal load, achieving dynamic load balancing. If a workstation goes down, the manager will detect the failure and assign unfinished tasks to other workstations.

This requires consideration of the granularity of the tasks. From a load balance point of view, the smaller the task size, the better the load balance. On the other hand, large number of tasks requires a lot of communication between the task manager and workers. So the task size should be chosen with both factors taken into account. Basically, if the speeds of the participating workstations do not differ greatly and the loads of the workstations are relatively even, larger task size (fewer number of tasks) should be used. If a variety of different workstations are used, and the loads of those workstations are changing frequently, smaller task size should be chosen to achieve dynamic load balancing and to minimize the overall time needed for the completion of all tasks.

Results and Performance Comparison

We compared the performances of the sequence comparison algorithm on a cluster and the intel iPSC/860 Hypercube. The cluster consists of one ALPHA workstation, five SUN SPARC 20

and two SUN SPARC 10 workstations. Sequences of different sizes were searched on the cluster and the Hypercube with different numbers of processors. The target database was the protein database SwissProt (Release 29), with 15.5 million bytes of sequences. For this set of workstations, the database was divided into 32 tasks, each with half million bases. The search returned the best matching scores as well as the positions where the best local alignments occurred. The tests were done with light load on the cluster. Times were measured using elapsed wall clock time which we believe reflected more realistically the actual application running time than CPU time. In Table 1 and 2, protein sequences of length 50, 100, and 150 are used as the query sequences.

# proc	len = 50	len = 100	len = 150
1	919	1806	2757
2	494	981	1458
4	260	515	770
8	141	271	387

Table 1: The times in seconds for the sequence comparison on the cluster.

# proc	len = 50	len = 100	len = 150
1	966	1856	2766
2	474	931	1386
4	241	471	702
8	124	242	349

Table 2: The times in seconds for the sequence comparison on the iPSC/860.

Conclusions

The development and implementation of a distributed sequence comparison algorithm based on the PVM system has been presented. The performance of the distributed algorithm is compared with that of the parallel implementation on an intel iPSC/860 Hypercube, and is shown to achieve comparable performance. Both search algorithms are used as sequence comparison tools for the GRAIL and GENQUEST Internet servers.

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*, 1990, vol. 215, 403-410.
- [2] A. Beguelin, J. Dongarra, A. Geist, R. Manchek, and V. Sunderam, "User's Guide to PVM: Parallel Virtual Machine," *ORNL/TM-11826*, 1991, Mathematical Sciences Section, Oak Ridge National Laboratory.

- [3] X. Guan, Richard Mural, Reinhold Mann, and Edward Uberbacher, "On Parallel Search of DNA Sequence Databases," *Proc., The Fifth SIAM Conference on Parallel Processing for Scientific Computing*, 1991, 332-337.
- [4] X. Guan, Richard Mural, Reinhold Mann, Ralph Einstein, and Edward Uberbacher, "GRAIL: An Integrated Artificial Intelligence System for Gene Recognition and Interpretation," *Proc., The Eighth IEEE Conference on Artificial Intelligence for Applications*, 1992, 9-13.
- [5] X. Guan, Richard Mural, Sergey Petrov, and E. C. Uberbacher, "A Sensitive Sequence Comparison Server for DNA and PROTEINS," *Proc., Genome Sequencing and Analysis Conference V*, Hilton Head Island, South Carolina, 1993.
- [6] O. Gotoh, "An Improved Algorithm for Matching Biological Sequences," *Journal of Molecular Biology*, vol. 162, 1982, 705-708.
- [7] W. R. Pearson, and D. J. Lipman, "Improved Tools for Biological Sequence comparison," *Proc. National Academy of Sciences, USA*, 1988, vol. 85, 2444-2448.
- [8] R. F. Doolittle, "Searching Through Sequence Databases," in Doolittle, R. F. (ed.) *Methods in Enzymology*, 1990, vol. 183, Academic Press, 99-110.
- [9] T. F. Smith, and M. Waterman, "Comparison of Biosequences," *Advances in Applied Mathematics*, 1981, vol. 2, 482-489.

94/09/07
14:14:24

cfp.ascii

9th INTERNATIONAL PARALLEL PROCESSING SYMPOSIUM (IPPS '95)

April 25-28, 1995 - Fes Parker's Red Lion Resort - Santa Barbara, California

Sponsored by IEEE Computer Society Technical Committee on Parallel Processing

In cooperation with:

ACM SIGARCH

Department of Computer Science, University of California, Santa Barbara

Institute for Advanced Computer Studies, University of Maryland, College Park

Information in the following order:

CALL FOR PARTICIPATION (Workshops, Tutorials, Industrial/Commercial)

CALL FOR PAPERS (Instructions, Program Committee)

ORGANIZATION (Organizers/Advisors, Location, Dates)

CALL FOR PARTICIPATION

The Technical Committee on Parallel Processing invites engineers and scientists from around the world to participate in the 9th International Parallel Processing Symposium. Established to advance knowledge in all areas of parallel processing and related technologies, the Symposium is the committee's primary event for presenting the most recent research and development findings. The now customary IPPS events include contributed technical papers, keynote speeches and panels, and workshops and tutorials.

WORKSHOPS: Workshops offer the opportunity for in-depth exploration of selected topics. Individuals and/or special interest groups are encouraged to submit proposals which cover a particular topic in the field. Send a one page proposal to the Symposium Chair by September 30, 1994.

SYMPOSIUM CHAIR

Viktor K. Prasanna

Dept. of EE-Systems, EEB 244

University of Southern California

Los Angeles, CA 90089-2562, USA

Vox 213-740-4483 / Fax 213-740-418

Internet: ipps95@halcyon.usc.edu

TUTORIALS: Proposals are solicited for organizing full or half-day tutorials. Interested individuals should submit a proposal to the Tutorials Chair by October 31, 1994. It should include a brief description of the intended audience, a lecture outline, and a vita for the lecturer.

TUTORIALS CHAIR

Ashfaq Khokhar

1398 CS Building

Department of Computer Science

Purdue University

West Lafayette, IN 47907-1398, USA

Vox 317-494-6530 / Fax 317-494-739

Internet: ashfaq@cs.purdue.edu

INDUSTRIAL TRACK/COMMERCIAL EXHIBITS: These paired activities will give manufacturers of commercial products the opportunity to both explain and demonstrate their wares. The Industrial Track will consist of original technical papers authored and presented by representatives from various companies (one paper per company). Topics of interest for papers include but are not limited to descriptions, comparisons, and/or evaluations of the design, implementation, and/or performance of an existing commercial system. Commercial Exhibits will allow vendors to display their products in an informal "walk-up and talk" setting. Companies are

encouraged to participate in both of these events. For more details, contact the Industrial-Commercial Chair by October 31, 1994.

INDUSTRIAL-COMMERCIAL CHAIR

John K. Antonio

School of Electrical Engineering

1285 Electrical Engineering Building

Purdue University

West Lafayette, IN 47907-1285, USA

Vox 317-494-6416 / Fax 317-494-6440

Internet: jantonio@ecn.purdue.edu

CALL FOR PAPERS

Authors are invited to submit manuscripts that demonstrate original unpublished research in all areas of parallel processing including development of experimental or commercial systems. Topics of interest include but are not limited to:

- Performance Modeling/Evaluation
- Signal & Image Processing Systems
- Parallel Implementations of Application Tasks
- Parallel Algorithms
- Parallel Architectures
- Neural Networks
- Interconnection Networks
- Parallel Languages
- Programming Environments
- Parallelizing Compilers
- Special Purpose Processors
- VLSI Systems
- Scientific Computing

To submit an original research paper, send six hard copies of your complete manuscript (not to exceed 15 single-spaced pages of text using 12 point type on 8.5 x 11 inch pages) to the Program Chair. References, figures, tables, etc. may be included in addition to the fifteen pages of text. Be sure to include your postal and e-mail addresses and telephone and fax numbers. All manuscripts will be reviewed. Manuscripts must be received by September 20, 1994. Submissions received after that or exceeding the length limit may be returned and not considered. Notification of review decisions will be mailed by December 30, 1994. Camera-ready papers are due January 27, 1995. Proceedings will be available at the Symposium. Fax or electronic submissions will not be considered.

PROGRAM CHAIR

Joseph Ja'Ja'

IPPS '95

Institute for Advanced Computer Studies

University of Maryland

College Park, MD 20742, USA

Internet: joseph@src.umd.edu

Fax: 301-314-9658

PROGRAM VICE-CHAIRS

ALGORITHMS

Guy Bluelloch, Carnegie Mellon University

APPLICATIONS

Vipin Kumar, University of Minnesota

ARCHITECTURE

Allan Gottlieb, New York University

SOFTWARE