2009 15th International Conference on Parallel and Distributed Systems

A Case of Parallel EEG Data Processing Upon a Beowulf Cluster

Yufeng Yao, Jinyi Chang, Kaijian Xia
School of Computer Science and Engineering
Changshu Institute of Technology
Changshu, Jiangshu, China
yufeng.yao1981@googlemail.com

***Abstract*—Electroencephalogram (EEG) data processing applications have become routine tasks in both bioscience and neuroscience research, which are usually highly compute and data intensive. In this paper, we present a parallel method to analyze the huge EEG data with a Beowulf cluster. Through an example of the synchronization measurement of multiple neuronal populations, the procedure of exploiting the parallelism of EEG data processing applications to achieve speed-up has been detailed. The experimental results indicate that the execution efficiency of EEG data processing can be improved dramatically using parallel and distributed computing techniques even with inexpensive computing platform.**

Keywords-Parallel and Distributed Computing; Bioinformatics; Neuroinformatics; Beowulf Cluster; MPI;

I. INTRODUCTION

Nervous systems and brain functions are enormously complex. Those related applications of bioinformatics and neuroinformatics are routinely compute and data intensive. In the past few years the grain of experimental data has become increasingly fine with the quick advances in experimental techniques for recording neural activities [1][5][15]. As a result, the density and the spatial scale of collected EEG data have been increasing dramatically [10]. The computational complexity of this type of applications is often prohibitively large for a normal centralized computer. A typical example is the synchronization measurement of neuronal populations [9].

The attempt of utilizing limited computing resources to support particularly complicated applications like EEG data processing will result in extremely poor execution performance and intolerable constraint onto the scale of problem which can be solved. There exists a pressing need for a method to tackle the computation bottlenecks that bioscience and neuroscience communities often face. It is desirable to utilize computing resources that are relatively economic and available to most researchers.

Parallel and distributed computing technology can distribute computation load of an application to multiple process units thus to improve computation performance by exploiting extra computing resources [16][17]. Distributed computing sounds a natural solution to the bottleneck problem of executing large neuroscience applications.

Various software platforms and standards have been developed for users in academia and industry. Amongst a variety of parallel and distributed computing platforms, the so-called Beowulf cluster technique can provides an inexpensive solution to high performance computing [2]. A Beowulf cluster is built with commodity-of-the-shelf (COTS) computers interconnected in the form of a private local area network (LAN). It often uses Linux as the operating system and supports message passing programs such as Message Passing Interface [12]. MPI is a message-passing application programmer interface commonly used to sustain high performance distributed programming over LANs. This feature is enabled with the help of a set of dedicated library routines.

This work aims to explore the viability of (1) using parallel distributed computing to significantly promote EEG data processing in bioscience and neuroscience studies and (2) achieving this goal using inexpensive and available computing resources. EEG data appear in the form of multi-channel time series with a large time scale. An application for analyzing EEG data normally employs the same algorithms to process different portions of long time series. This type of applications inevitably possesses high degree of parallelisms which can be either spatial or temporal or both.

A typical example can be seen in [11](Li et al, 2007a). Li et al have introduced a synchronization measurement approach of multiple neuronal populations, which adopts a surrogate resampling method to quantify and describe properties of synchronization of neuronal populations. The approach was modeled using Matlab, and the data analysis has been performed upon a standalone desktop computer. The execution efficiency of the model is very low due to the heavily iterative processing on a huge EEG data set. In this study, an alternative parallel method has been designed to accelerate the application upon our dedicated Beowulf cluster.

The rest of this paper is organized as follows: Section 2 briefs the uses of parallel and distributed computing technologies for dealing with huge data sets in bioinformatics and neuroinformatics. The dedicated Beowulf cluster is detailed in Section 3. Section 4 introduces the algorithms for parallel synchronization measurement. Section 5 presents the experimental results and discussions. Section 6 concludes this work with a summary and proposals on future work.

II. RELATED WORK

In neuroinformatics and bioinformatics, how to effectively manage and analyze complex and massive experimental data remains a grand challenge [8]. An attempt has been made to improve the management of massive

1521-9097/09 $26.00 © 2009 IEEE
DOI 10.1109/ICPADS.2009.65

neuroimaging data [7] by integrating database management systems (DBMS) with Grid computing and cluster-based computing [6]. Their approach allows fMRI data processing to be accelerated by processing different parts of the data simultaneously over multiple distributed databases.

A database search program, namely DSEARCH, has been presented in [9]. The program aims to improve the existing searching algorithms for similar sequences in bioinformatics by parallelizing a search process across multiple processes. DSEARCH effectively enables bioinformatics researchers to benefit from the sensitivity of these algorithms while achieving significantly enhanced performance.

Grid computing was introduced in mammographic screening [3]. The approach grants the availability of a distributed database of mammograms from hospital in the virtual organization. Mammograms can be acquired in any location, and the related information required to select and access them can be queried by any client. The results of a query can be used by analysis algorithms executing on remote nodes to the user.

In [13], a distributed computing system has been designed for multivariate time series analyses of multichannel neurophysiological data. The system supports multiple measures within a single job and exhibits encouraging performance in analyzing multi-channel neural data. The applications based on the system perform equally well in a dedicated and a random access computer environment.

Speedup is the priory objective in bioinformatics and neuroinformatics application/systems using distributed computing. Distributed computing applies to a variety of applications with different forms of parallelism utilized. Both distributed computing technologies suitable for coarse-grained (e.g., Grid computing over Internet) and those suitable for fine-grained (e.g., computing supported by local area network and clusters) have successful applications. The results from the existing works indicate that it is feasible to integrate parallel and distributed computing with bioinformatics and neuroinformatics. This research area is still relatively young, and many open research issues are still very much in flux due to the complexity of the applications and the heterogeneity between distributed computing and bioscience and neuroscience.

III. THE PARALLEL AND DISTRIBUTED COMPUTING PLATFORM

A Beowulf cluster has been constructed as a dedicated parallel and distributed computing platform for EEG data processing. EEG data processing can be performed with the support of a customized computing engine.

A. Configuration of the Beowulf Cluster

Table I lists the configuration details of our Beowulf cluster. The cluster consists of a number of desktop PC nodes. Each node has 2G bytes of memory and an Intel Duo CPU (at 2.8GHz) while connectivity is provided by a 1Gbit Ethernet switch. The layout of the cluster can be illustrated in Figure 1. The cluster is managed by a master node equipped with two network interface cards (NICs), with eth1 and eth0 connecting to the Internet and the internal private Giga byte network respectively. The compute nodes are only connected the private network, and those perform the compute tasks assigned by the master node. The master node may also take compute tasks in the same manner as other nodes do.

The cluster operates with the support of a stable Linux operating system ClusterKnoppix [4]. ClusterKnoppix uses the V2.4 debian kernel, and it enables OpenMosix to ensure auto-detect of compute nodes. High performance communication is facilitated by LAM/MPI [12]. It provides a number of options for MPI communication with very little overhead. The TCP communication system provides near-TCP stack bandwidth and latency, even at Gigabit Ethernet speeds. Two shared memory communication channels are available, each using TCP for remote-node communication.

TABLE I. CONFIGURATIONS OF EACH NODE

CPU	Intel® Core™2 Duo E7400, 2.80 GHz
RAM	2G
Networking	1Gbps
Operating System	ClusterKnoppix V3.6
Message Passing Infrastructure	LAM/MPI 7.1.2

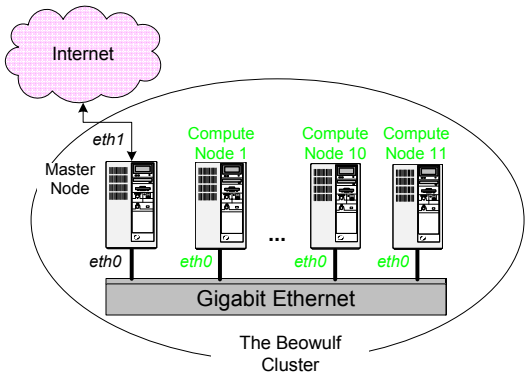

Figure 1. The Beowulf cluster and its components

B. The Parallel and Distributed Computing Engine

To cater for the needs of EEG data processing, a parallel and distributed computing engine has been developed upon the Beowulf cluster. The computing engine serves as a middleware between the data processing applications and the underlying cluster platform, which is illustrated in Figure 2.

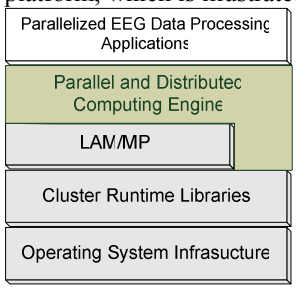

Figure 2. The parallel and distributed computing engine

The computing engine provides the following functionalities to the EEG data processing applications on its top:

- Computation task management. The master node uses this functionality to manage the resources of the compute nodes (including itself). Computation tasks defined in the above EEG data application will be created and assigned by the master node to all processing units in the cluster. On completion of a batch of tasks, the previous occupied computing resources may be freed.
- Synchronization amongst a group of tasks. Tasks will be grouped which represent an individual complete operation of the algorithms in the application. On completion of the submitted tasks, the same group of tasks will be synchronized to retrieve and process the returning data to ensure global state consistency.
- Runtime library of the Random Matrix Theory [7]. The library consists of a set of routines to support the calculations in Random Matrix Theory. This is a parallel implementation using C++ on top of the LAM/MPI.

IV. PARALLEL SYNCHRONIZATION MEASUREMENT

This section introduces a case concerning huge EEG data processing, i.e., synchronization measurement of neuronal populations. A parallelized synchronization measurement alternative has been designed.

A. The Original Synchronization Measurement Approach

Neuronal population synchronization is an important feature of normal and abnormal brain function. Estimating the degree of synchronization within and between neuronal populations of varying sizes is an approach to synchronization measurement [10]. The approach proposed by Li et al [11] includes two steps.

Step1: The approach estimates correlation between all pairs of signals, to produce a correlation matrix from a multivariate neuronal activity time series (EEG data as a multiple channel time series). The whole chunk of EEG data was gradually processed per time window t_k, where $i = 1, ..., M$ (the channel number), $k = 1, ..., T$ (the number of data points in the time window). An equal time correlation matrix **C** will then be constructed. The eigenvalues of **C** will provide information about the synchronization between individual elements of the matrix and the highest eigenvalue can give information about global synchronization.

Step 2: A Correlation Matrix Analysis method to analyze the correlation matrix based on the Radom Matrix Theory. The initial objective is to derive a global synchronization index to obtain a normalized of global synchrony value independent of the number of channels (M). The method needs to randomize all channels of the time series by N (the times of surrogating, which should be large enough) using Amplitude Adjusted Fourier Transform [14] to destroy each existing equal time correlation.

A correlation matrix **R** for the surrogated time series will be computed, and the global synchronization index can be calculated as follows. A synchronization index equal to 0 or 1 denotes no synchrony or perfect synchrony among the time series. The eigenvalues of **R** can educe the number of clusters of synchronization. An eigenvalue indicates the strength of a cluster, and the corresponding eigenvector describes its internal structure.

B. The Parallel Synchronization Measurement Approach

The parallelism of the synchronization approach is both temporal and spatial:

Temporal parallelism: After the time series are normalized, they will be randomized for **a large number of times** to get the surrogated correlation matrix based on which the synchronization index can be obtained. This procedure is compute intensive. For a single piece of data in a time window, randomization calculations operate independently from each other. Statistics will be made only after results collected from each randomized time series are consolidated. The degree of temporal parallelism of the program is equal to the number of times of surrogating, which is very high.

Spatial parallelism: Multiple main iterations are performed based on the number of EEG data segments. Inside a main iteration, the program performs a band-pass filtering on the raw EEG data using harmonic wavelet transform. Subsequently, the filtered EEG data series are to be processed based on a sliding time window. It should be noted that a main iteration is executed specifically on a piece of EEG data, and the iteration does not rely on data generated from other iterations or interoperate with other iterations when performing the heavy computation described as above. The degree of parallelism of spatial program is equal to the number of raw EEG data segments.

The above facts imply a large space to optimize the execution of the synchronization measurement approach using parallel and distributed computing. Based on the features the existing parallelisms, two major re-engineering works were performed: (1) a parallel implementation of the main iteration, which may operate independently on one or more EEG data segments; and (2) an individual routine for randomization of EEG data segment to support parallel surrogating, and these routines can be performed and coordinated on multiple compute nodes.

After all, the overall distributed program only incurs minimal revision to the original approach [11]. A performance comparison between the two implementations is given in Section V.

V. EXPERIMENTS AND RESULTS

The neuronal population activities were simultaneously recorded with 32 electrodes. In total, there are raw EEG data for 7200 Seconds' recordings of neuronal population activities. The EEG data consist of multiple extracellular field potentials measured in vitro from the hippocampus of male Sprague-Dawley rats. The field potentials are produced by the synchronous activity in populations of neurons close to each electrode.

The same batch of EEG raw data segments have been used for analysis by the two versions of programs. Each raw data segment records the extracellular field potentials over

10 Seconds, and there are 720 EEG data segments. The time window size is set to 0.2 Second and the time step is set to 0.1 Second. With the sampling frequency 1e+004, a window covers 2000 sampled data points for each channel.

The synchronization measurement approach is built into two different versions in C++: (1) the original sequential version and (2) the parallel version. The master node hosts the client session. The sequential synchronization measurement was executed at the master node, and it in average took about 13.7 Seconds per data segment and about 9864 Seconds to complete processing the whole data. Two sets of experiments had been performed using the parallel alternative over the Beowulf cluster.

A. Varying the number of compute nodes

In the first set of experiments, each node took 8 surrogating tasks in an iteration with each iteration processing 2 data segments at one time. The experimental results are presented in Figure 3 in terms of execution time.

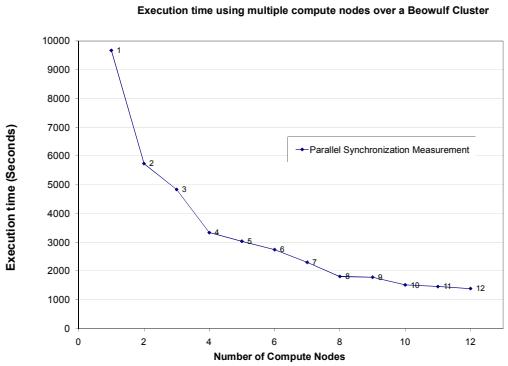

Figure 3. Execution time with different number of compute nodes

The execution time decreases rapidly with the number of compute nodes. The execution time is 9672 Second (slightly lower than the original synchronization measurement result) when one node was used for executing the parallel synchronization measurement. In the case of two nodes, the execution time goes down to about 5736 Seconds (about 59% of the case of one node supporting the parallel program's result and about 55% of the original program's result). The execution efficiency of the parallel synchronization measurement tends towards flat with the increasing number of compute nodes when more than 8 nodes were used. The encouraging results indicate that the parallel computing can significantly improves the execution efficiency and the parallel implementation scales very well.

B. Varying the Number of Data Segments per Iteration

In the second set of experiments, 5 compute nodes were used with each node taking 8 surrogating tasks in an iteration. Each iteration may process different number of data segments (varied length of EEG recordings) at one time. The number of data segments are varied from 2 to 20. The experimental results are shown in Figure 4.

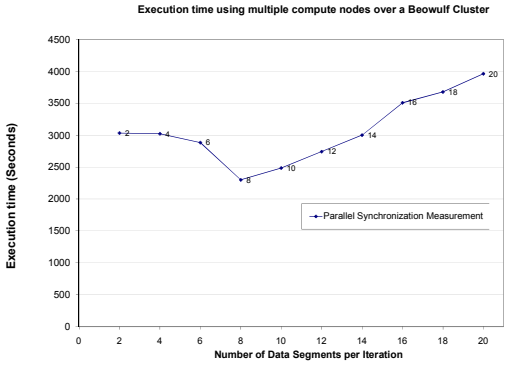

Figure 4. Execution time with different number of data segments per iteration

At the beginning, the execution time decreases the increasing number of data segments. The execution time becomes about 2301 Seconds (about 76% of the starting case) with an iteration processing 8 data segments. With more data segments being processed per iteration, the execution time increases. The results indicate that optimal execution performance may be obtained only when parallelism is properly exploited. The results well comply with the Amdahl's law.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we present an example of incorporating parallel and distributed computing technology into EEG data processing applications. Such applications normally possess a high degree parallelism. To improve the execution performance of this type of applications thus to promote the research in bioscience and neuroscience research, we have developed a cost-effective alternative to large supercomputers.

A Beowulf cluster has been constructed as a dedicated parallel and distributed computing platform for EEG data processing. To meet the needs of EEG data processing, a parallel and distributed computing engine has been developed upon the Beowulf cluster. The computing engine operates as a middleware interfacing the data processing applications and the underlying cluster platform. The computing engine supports computation task management, synchronization amongst a group of tasks, and routines to support calculations on Random Matrix theory. The light weighted engine encapsulates the LAM/MPI and persists in the form of a C++ library.

A case of synchronization measurement of neuronal populations was analyzed. The method aims to quantify and describe properties of synchronization of population neuronal activities recorded simultaneously from multiple sites. A problem of the method is its poor execution

performance. Based on its feature of parallelism, a parallel approach to synchronization measurement has been designed.

The parallel synchronization measurement approach has successfully tackled this problem by exploiting the high degree of temporal and spatial parallelism of the original sequential implementation. The parallel synchronization measurement approach incurs only minimal reengineering effort. Experiments have been performed (1) to compare the performance of two alternative approaches to synchronization measurement, and (2) to evaluate the efficiency of the parallel and distributed computing platform. The experimental results show that the proposed distributed approach can dramatically reduce the execution time compared with the computing resources of the Beowulf cluster. It can be observed that the parallel computing can significantly improves the execution efficiency and the parallel implementation scales very well, and it can provide very effective solutions to poor performance when the degree of parallelism is high. The results also indicate that optimal execution performance may be obtained only when parallelism is properly exploited.

For our future work, we need to investigate the feasibility of using Grid services to support a Service oriented Architecture to allow the platform serving the needs of both fine-grained and coarse-grained EEG data processing applications.

REFERENCES

[1] Baruchi I, Volman V, Raichman N, Shein M, Ben-Jacob E. The Emergence and Properties Of Mutual Synchronization in In Vitro Coupled Cortical Networks. Eur J Neurosci 28(9):1825-35, 2008.

[2] Beowulf, http://www.beowulf.org, 2009.

[3] Cerello P, Bagnasco S, Bottigli U, Cheran Sc, Delogu P, Fantacci Me, Fauci F, Forni G, Lauria A, Lopez Torres E, Magro R, Masala Gl, Oliva P, Palmiero R, Ramello L, Raso G, Retico A, Sitta M, Stumbo Skoo, Tangaro S, Zanon E. Gpcalma: A Grid-Based Tool for Mammographic Screening. Methods Inf Med. 2005;44(2):244-8.

[4] ClusterKnoppix. http://clusterknoppix.sw.be/, 2009.

[5] Erickson J, Tooker A, Tai Yc, Pine J. Caged Neuron Mea: A System For Long-Term Investigation Of Cultured Neural Network Connectivity. J Neurosci Methods 175(1):1-16, 2008.

[6] Globus, http://www.globus.org/toolkit/, 2009.

[7] Hasson U, Skipper Ji, Wilde Mj, Nusbaum Hc, Small Sl. Improving The Analysis, Storage And Sharing Of Neuroimaging Data Using Relational Databases And Distributed Computing. Neuroimage 39(2):693-706, 2008.

[8] Jeffrey L. Teeters, Kenneth D. Harris, K. Jarrod Millman, Bruno A. Olshausen And Friedrich T. Sommer, Data Sharing For Computational Neuroscience, Neuroinformatics. 6(1):47-55, 2008.

[9] Keane, T. M., Naughton, T. J. Dsearch: Sensitive Database Searching Using Distributed Computing. Bioinformatics 21(8):1705-1706, 2005.

[10] Le Van Quyen M, Bragin A. Analysis Of Dynamic Brain Oscillations: Methodological Advances, Trends Neurosci, 30: 365-373, 2007.

[11] Li Xl, Dong Cui, Premysl Jiruska, John E Fox, Xin Yao, And John Gordon Ralph Jefferys, Synchronization Measurement Of Multiple Neuronal Populations Based On Correlation Matrix Analysis, Journal Of Neurophysiology (October 3, 2007). Doi:10.1152/Jn.00977.2007

[12] LAM/MPI, http://www.lam-mpi.org/, 2009.

[13] Müller, A, Osterhage, H., Sowaa, R., Andrzejakc, R. G., Mormanna, F., Lehnertz, K., A Distributed Computing System For Multivariate Time Series Analyses Of Multichannel Neurophysiological Data, Journal Of Neuroscience Methods, Volume 152, Issues 1-2, 15 April 2006, Pages 190-201.

[14] Schreiber T, Schmitz A. Improved Surrogate Data For Nonlinearity Tests. Phys Rev Lett 77: 635-638, 1996.

[15] Stieglitz T. Neural Prostheses In Clinical Practice: Biomedical Microsystems In Neurological Rehabilitation. Acta Neurochir Suppl. 2007;97(Pt 1):411-8.

[16] Wang, L., Marcel Kunze, Jie Tao. 2008. Performance evaluation of virtual machine-based Grid workflow system. Concurrency and Computation: Practice and Experience 20(15): 1759-1771.

[17] Wang, L., Wei Jie. 2009. Towards supporting multiple virtual private computing environments on computational Grids. Advances in Engineering Software 40(4): 239-245.