A Massively Parallel Approach for Nonlinear Interdependency Analysis of Multivariate Signals with GPGPU

Dan Chen
School of Computer Science
China University of Geosciences
Wuhan, China
E-mail: chendan@pmail.ntu.edu.sg

Lizhe Wang
Center for Earth Observation and Digital Earth
Chinese Academy of Sciences
Beijing, China
E-mail: lzwang@ceode.ac.cn

Dong Cui, Dongchuan Lu, Xiaoli Li
Institute of Electrical Engineering
Yanshan University
Qinhuangdao, China
E-mail: xiaoli.avh@gmail.com

Samee U. Khan
Electrical & Computer Engineering Dept
North Dakota State University
ND, USA
E-mail: samee.khan@ndsu.edu

Joanna Kołodziej
Department of Mathematics and Computer Science
University of Bielsko-Biała
Poland
E-mail: jkolodziej@ath.bielsko.pl

Abstract—**Nonlinear interdependency (NLI) analysis is an effective method for measurement of synchronization among brain regions, which is an important feature of normal and abnormal brain functions. But its application in practice has long been largely hampered by the ultra-high complexity of the NLI algorithms. We developed a massively parallel approach to address this problem. The approach has dramatically improved the runtime performance. It also enabled NLI analysis on multivariate signals which was previously impossible.**

Keywords-GPGPU; synchronization; nonlinear interdependency; EEG; massively parallel computing

I. INTRODUCTION

Understanding of the mechanisms underlying synchronization between neuronal populations is extremely important in the study of normal and abnormal brain functions. Measures of synchronization are important and challenging in the study of neuronal dynamics. Synchronization phenomena have been increasingly recognized as a key feature for establishing the communication between different regions of the brain [10][16].

There exist numerous approaches and measures for characterizing synchronizations of bivariate data, and those typically include cross-correlation, mutual information, spectrum-based coherence, nonlinear interdependency (NLI), and phase synchronization [1][9][11][17]. Amongst those, the NLI analysis approaches are salient. NLI is a measure of generalized synchronization in nonlinear systems. It measures the interdependency according to the distance of delay vectors in two systems. An NLI approach can not only compute the synchronization strength but also indicate the synchronization direction, i.e., which system (driver) drives the other system (response).

However, applications of NLI approaches in practice have long been hampered by the ultra-high complexity of the NLI algorithms [7]. More details about NLI are available in Section III. The complexity increases in the order of N^2 with the number of data channels. For example, it typically takes weeks to process a 10-channel electroencephalogram (EEG, a typical signal as a signature of neural activities) segment on top of a high-end desktop.

As experimental techniques for recording neural activities have been advancing quickly, i.e., rapidly increasing number of channels (electrodes) and sampling frequencies. The density and the spatial scale of neural signals have been increasing exponentially, i.e., in hundreds and more than 10KHz. Neural signal analysis with NLI manifests a highly compute intensive problem. There is a pressing need for an improved NLI approach (1) to be able to analyze more data channels thus to better understand the synchronization mechanism among brain regions in a larger scale and (2) to minimize the cycle of analysis so that timely decisions may be made in reaction to the changing status of patients in real life.

There exists high built-in parallelism in the NLI algorithm. It sounds a natural choice to develop a parallel NLI approach on a high performance computing cyberinfrastructure [6] to fulfill the performance and scalability requirement by utilizing the multi-level parallelisms in an application of NLI-based synchronization analysis.

The many-core computing platform (e.g., graphics processing unit) provides a massively parallel computing platform [3][4]. We have developed a parallel NLI approach, which successfully adopts general-purpose computing on the graphics processing unit (GPGPU). Our approach properly parallelizes the NLI analysis application, in particular the sorting operations and manipulation of matrices. The proposed approach has been implemented using NVIDIA's CUDA, a general-purpose parallel computing architecture. The intensive computation involved in the whole problem space is then mapped onto a large number of threads evenly executed by the many parallel cores of GPU(s).

We presented a case study of employing NLI to analyze a multivariate EEG dataset generated by an epilepsy patient seizures occurred. We carried out performance study on the GPGPU-enabled parallel approach. For an EEG segment with length of an hour, the parallel approach can complete analysis in less than an hour while the original serial counterpart needs hundreds of hours to handle the same task. The results for synchronization measurement are also meaningful for understanding the directional synchronization amongst multiple brain regions which was previously impossible.

The rest of this paper is organized as follows: Section II introduces related work in uses of high performance computing techniques for neuroscience problems. Section III details the NLI application and analyzes its parallelisms. The parallel NLI approach is presented in Section IV. Experiments and results are available in Section V. Section VI concludes the paper with a summary and proposal for future work.

II. Related Work

Recently, more and more researchers started to use high performance computing techniques to address neuroscience problems. Here we highlight several typical works along this direction:

In [12], a distributed computing system is introduced for multivariate time series analyses of multivariate neurophysiologic data. The system supports multiple measures within a single job and attempts to facilitate a general environment to enhance performance in analyzing multivariate neural data.

In order to process massively electrocorticographic (ECoG) signals [18], Wilson and Williams used GPGPU to parallelize the processing algorithms. Although encouraging results have been obtained, their approach focuses on accelerating matrix-matrix multiplication and auto-regressive method. The two target methods are widely used in other research areas and disciplines which are not specific to neural data analysis.

Another interesting work is the efficient parallel algorithms for fitting arbitrary spiking neuron models to electrophysiological data [13]. This computational technique relies on vectorization and parallel computing techniques to achieve efficiency via the use of GPGPUs. It can be used by modelers in systems neuroscience to obtain empirically validated models for their studies.

Our group has developed a GPGPU-enabled approach (1) to decompose a complex non-linear and non-stationary signal into a number of components each carrying the true physical characteristics in a frequency, and (2) to compute the spectral entropy thus to monitor the evolving dynamics of the signal. This work does not concern the relationship amongst different data channels [3].

This study is one of the first work that exploit many-cores architectures to measure synchronization amongst neural signals in multiple individual channels. The proposed approach targets on applications in practice so that it needs (1) to minimize the cycle of massive EEG data analysis, (2) to analyze synchronization amongst neuronal activities generated from brain regions in a relatively large scale, thus timely and accurate decisions may be made, e.g., accurate localization of epileptic focus. This study explores the feasibility of achieving satisfactory performance with inexpensive and available cyberinfrastructures [19][20][21][22].

III. The Nonlinear Interdependency Application

In this section, we introduce the NLI algorithm and the application for NLI analysis. The parallelisms embedded in the application are discussed.

A. *Algorithm for Nonlinear Interdependency Analysis*

First, we recap the basic algorithm for NLI analysis, which operates on two arrays (or time series, EEG data channels in the context of this study). Let $X = (x_1, x_2, \ldots, x_T)$, $Y = (y_1, y_2, \ldots, y_T)$ denote two time series measured from two systems (e.g., brain sub-regions), T is the length of the time series. The flow of the algorithm is presented in Figure 1.

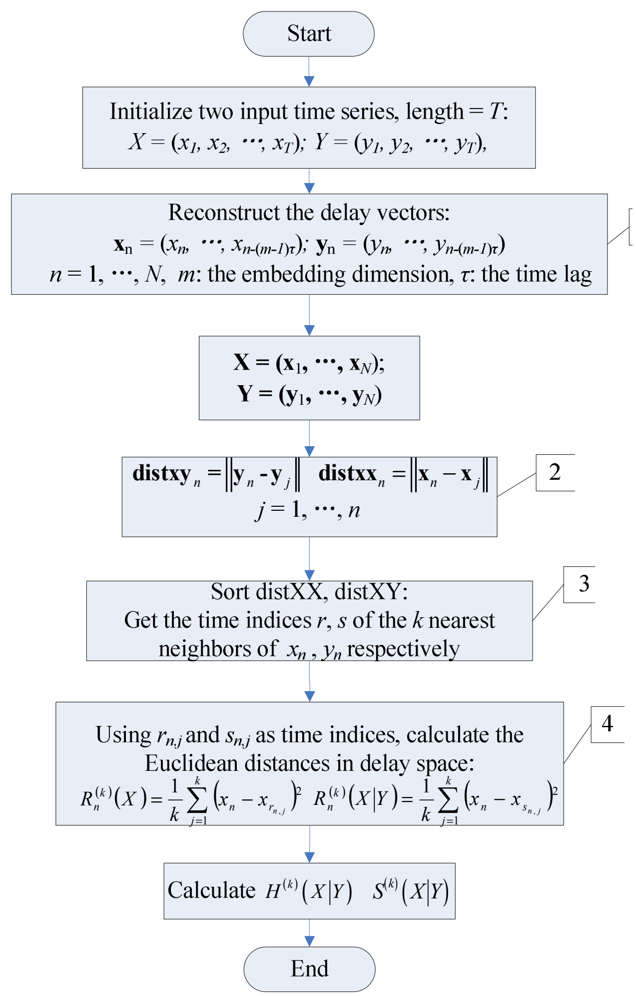

Figure 1. A basic NLI algorithm.

- Step 1 reconstructs the delay vectors [14]: Time lag embedding in an m-dimensional phase-space leads to phase-space vectors $\mathbf{x}_n = (x_n, \ldots, x_{n-(m-1)\tau})$; $\mathbf{y}_n = (y_n, \ldots, y_{n-(m-1)\tau})$, where $n = 1, \cdots, N$, m is the embedding dimension and τ denotes the time lag. Let's define $N = T-(m-1)\tau$, for delay vectors' indices: $1 \leq n \leq N$. Matrices with the arrays of delay vectors are then constructed: $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_N)$ and $\mathbf{Y} = (\mathbf{y}_1, \cdots, \mathbf{y}_N)$.
- Step 2 calculates distances: The distance between any two delay vectors in **X** (and **Y**) is computed. Let's define $\mathbf{distxx}_n = \|\mathbf{x}_n - \mathbf{x}_j\|$ and $\mathbf{distxy}_n = \|\mathbf{y}_n - \mathbf{y}_j\|$, where $j = 1, \cdots, n$. Eventually, we have $\mathbf{distXX} = (\mathbf{distxx}_1, \mathbf{distxx}_2, \ldots, \mathbf{distxx}_n)$ and $\mathbf{distXY} = (\mathbf{distxy}_1, \mathbf{distxy}_2, \ldots, \mathbf{distxy}_n)$.
- Step 3 sorts distances: For results in **distXX** and **distXY**, this step sorts them in an ascending order. The k nearest neighbors of $\mathbf{x}_n$ and $\mathbf{y}_n$ are identified, and their time indices are written as $r_{n;j}$ and $s_{n;j}$, $j = 1; \ldots; k$.
- Step 4 calculates the Euclidean distances in delay space: For each $\mathbf{x}_n$, distances between vectors can be written as: $d(\mathbf{X})_n^{(1)} \equiv \left\|\mathbf{x}_n - \mathbf{x}_{r_{n,1}}\right\| = \min_q \left\|\mathbf{x}_n - \mathbf{x}_q\right\|$, $d(\mathbf{X})_n^{(2)} \equiv \left\|\mathbf{x}_n - \mathbf{x}_{r_{n,2}}\right\| = \min_{q \neq r_{n,1}} \left\|\mathbf{x}_n - \mathbf{x}_q\right\|$, …, In the delay space, where $\left\|\mathbf{x} - \mathbf{x}'\right\|$ is the Euclidean distance in delay space. For each $\mathbf{x}_n$, the squared mean Euclidean distance to its k closest neighbors is define as $R_n^{(k)}(\mathbf{X}) = \frac{1}{k}\sum_{j=1}^{k}\left(\mathbf{x}_n - \mathbf{x}_{r_{n,j}}\right)^2$ and the Y-conditioned mean squared Euclidean distance is defined by replacing the nearest neighbors by the equal time partners of the closest neighbors of $\mathbf{y}_n$, $R_n^{(k)}(\mathbf{X}|\mathbf{Y}) = \frac{1}{k}\sum_{j=1}^{k}\left(\mathbf{x}_n - \mathbf{x}_{s_{n,j}}\right)^2$. Similarly, one can define $R_n^{(k)}(\mathbf{Y}) = \frac{1}{k}\sum_{j=1}^{k}\left(\mathbf{y}_n - \mathbf{y}_{r_{n,j}}\right)^2$ and $R_n^{(k)}(\mathbf{Y}|\mathbf{X}) = \frac{1}{k}\sum_{j=1}^{k}\left(\mathbf{y}_n - \mathbf{y}_{s_{n,j}}\right)^2$.

Based on the above definitions, one has an interdependence measure $S^{(k)}(\mathbf{X}|\mathbf{Y})$ as: $S^{(k)}(\mathbf{X}|\mathbf{Y}) = \frac{1}{N}\sum_{n=1}^{N}\frac{R_n(\mathbf{X})}{R_n^{(k)}(\mathbf{X}|\mathbf{Y})}$, where $R_n(\mathbf{X})$ is the average squared radius of the point cloud $\{\mathbf{x}_n\}$:. $R_n(\mathbf{X}) = (N-1)^{-1}\sum_{j \neq n}(\mathbf{x}_n - \mathbf{x}_j)^2$. Low values of $S^{(k)}(\mathbf{X}|\mathbf{Y})$ indicate independence between systems **X** and **Y**, while high values indicate synchronization. One can define another non-linear interdependence measure $H^{(k)}(\mathbf{X}|\mathbf{Y})$ as $H^{(k)}(\mathbf{X}|\mathbf{Y}) = \frac{1}{N}\sum_{n=1}^{N}\log\frac{R_n(\mathbf{X})}{R_n^{(k)}(\mathbf{X}|\mathbf{Y})}$. It is zero if **X** and **Y** are completely independent, while it is positive if closeness in **Y** implies closeness in **X** for equal time partners. It would be negative if close pairs in **Y** corresponded mainly to distant pairs in **X**. The H measure is more resistant to noise than S. One can have more measures $N^{(k)}(\mathbf{X}|\mathbf{Y}) = \frac{1}{N}\sum_{n=1}^{N}\frac{R_n(\mathbf{X}) - R_n^{(k)}(\mathbf{X}|\mathbf{Y})}{R_n(\mathbf{X})}$ and $M^{(k)}(\mathbf{X}|\mathbf{Y}) = \frac{1}{N}\sum_{n=1}^{N}\frac{R_n(\mathbf{X}) - R_n^{(k)}(\mathbf{X}|\mathbf{Y})}{R_n(\mathbf{X}) - R_n^{(k)}(\mathbf{X})}$. The N and M measures are even more robust.

A very important feature of these measures is asymmetry, e.g., one normally has $S^{(k)}(\mathbf{X}|\mathbf{Y}) \neq S^{(k)}(\mathbf{Y}|\mathbf{X})$. If $S^{(k)}(\mathbf{X}|\mathbf{Y}) > S^{(k)}(\mathbf{Y}|\mathbf{X})$, i.e., **X** depends more on **Y** than vice versa, we say that **Y** is more 'active' than **X**. Hence the direction of synchronization, i.e., which system is the driver in **X** and **Y**, can be definitely identified through the measures.

We call the operation of the basic NLI algorithm on a pair of time series an NLI procedure in the rest of this paper.

B. A Synchronization Measurement Application with NLI for Multivariate EEG

This study extends the basic NLI algorithm to enable analyzing multivariate EEG dataset with NLI synchronization measures. Given M data channels, for the ith data channel $\mathbf{X}_j$ (also written as **Y**), we define a global NLI measure based on the concept of S measure (i.e., $S^{(k)}(\mathbf{X}|\mathbf{Y})$) to quantify the synchronization effect of **Y** to all other data channels:

$$S^{(k)}(\{\mathbf{X}\}|\mathbf{Y}) = \sum_{i=1, i\neq j}^{M} S^{(k)}(\mathbf{X}_i|\mathbf{Y}).$$

Similarly, we define another global NLI measure to quantify the synchronization effect of other data channels to **Y**:

$$S^{(k)}(\mathbf{Y}|\{\mathbf{X}\}) = \sum_{i=1, i\neq j}^{M} S^{(k)}(\mathbf{Y}\,|\,\mathbf{X}_i).$$

If $S^{(k)}(\{\mathbf{X}\}|\mathbf{Y}) > S^{(k)}(\mathbf{Y}|\{\mathbf{X}\})$, we consider that **Y** affects other data channels more and **Y** is more active.

Assume the EEG dataset consists of M time series recorded simultaneously, we apply a sliding time window to these EEG time series, corresponding to M channels, to generate numerous M short time series (epochs) covered by the window. There exists an overlap of a fixed length between two consecutive time windows. The application operates as depicted in Figure 2.

- Step 1 initializes the application: The size of the sliding time window is set to *Epoch*; the overlap is set to *Overlap*; the number of windows to cover the whole EEG dataset is obtained as N = length (EEG) - *Epoch*/(*Epoch-Overlap*); the offset of the slide time window, t, is set to 1; the indices of dual data channels to be selected, i and j, are set to 1.

- Step 2 performs the basic NLI analysis on each pair of epoch in the current time window: for any epoch in the time window, this step calculates the NLI measures against the other (M-1) epochs. This step continues until all combinations of epoch pairs are processed. The slide window moves the next offset until the whole EEG data is analyzed.
- Step 3 calculates the S-estimator [1][15]: This step first calculates the eigenvalues and eigenvectors of the correlation matrix composed by NLI measures obtained from step 2. Afterwards, the global NLI measures and the S-estimator can be obtained to quantify how much the data channels are correlated.

C. Parallelisms at Multiple Levels

From the introductions to the basic NLI algorithm and the application of global synchronization measurement with NLI for multivariate EEG (see Sections III.*A* and III.*B*), we can clearly identify the embedded parallelisms in the synchronization application. The parallelisms exist in at least two levels, namely **the application level** and **the NLI level**:

- The application level. The NLI procedure is treated as a whole at this level. For a time window, the NLI procedures are independent before calculating the global NLI measures. An NLI procedure can complete without inputs from any other NLI procedures.
- The NLI level. Inside an individual NLI procedure, there exist numerous computations which may be performed concurrently: (1) reconstruction of the phase space using delay vectors on different data channels is independent; (2) calculation of the distance between any two delay vectors in a reconstructed matrix is independent; and (3) sorting the delay vector distances may be parallelized.

We can parallelize the NLI application by naturally adapting to its parallelism at the application level and/or the NLI level. Obviously, the computations at the NLI level may be handled by tasks with a much finer grain than that at the NLI level (see Fig. 7). Tasks at this level suit more for a platform with fine-grain parallelisms such as a GPU or a many-cores computer.

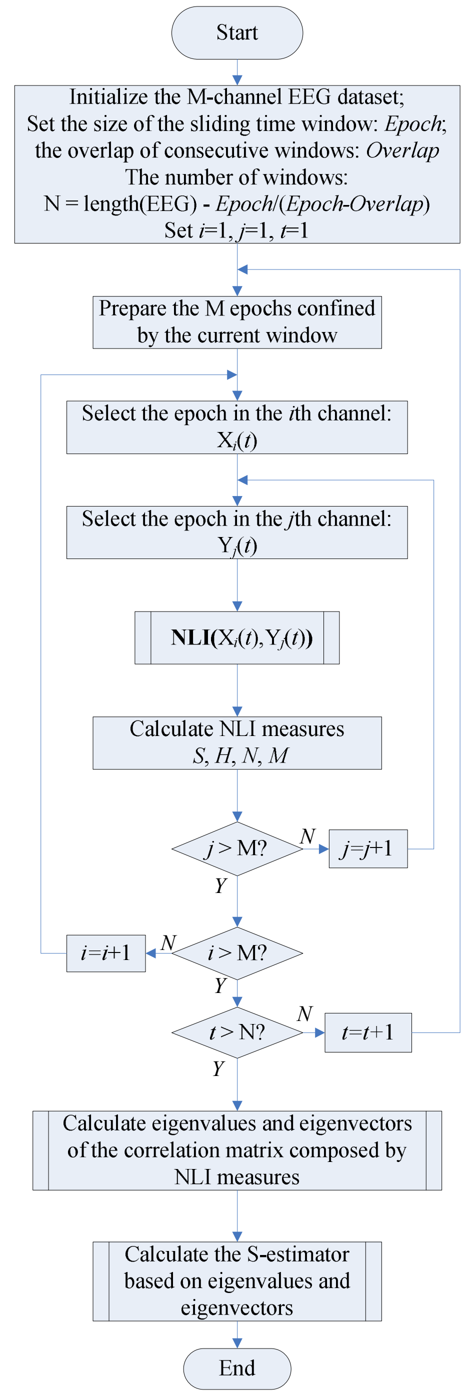

Figure 2. The extended NLI analysis application for multivariate EEG.

IV. Parallel Nonlinear Interdependency Analysis with GPGPU

We have developed a massively parallel NLI approach on top of CUDA by adapting to the NLI level parallelism.

A. *Massively Parallel Computing Using GPGPU*

A modern GPU possesses single-instruction multiple-data (SIMD) cores, high on-chip bandwidth, and explicit data transfers between fast local memories and external dynamics RAM. All of these establish GPU a prior option when coping with data and compute intensive problems [3][4].

GPGPU has recently boomed with the enhanced programmability of GPUs. Relatively high abstraction levels have emerged, and NVIDIA's Compute Unified Device Architecture (CUDA) is salient. The current CUDA fosters a software environment, which allows programming on a GPU with a slightly extended version of C, a CUDA-based application can execute on the many cores of GPU(s) in a massively parallel fashion.

A CUDA-based application defines C functions as "kernels" to be explicitly executed on the GPU (referred to as the "device"). Normally the kernels are invoked from the main executable of the application on the CPU (the "host''), which will be executed for N times via N individual CUDA "threads". CUDA threads are grouped to blocks, and threads in a block can cooperate together by efficiently sharing data through a fast shared memory and synchronizing their execution to coordinate memory access. Threads, memories and synchronization are exposed to developers for very fine-grained data and thread parallelism, e.g., dealing with a single float type variable per thread. A developer should partition the problem into a set of fine-grained subtasks, which can be mapped to CUDA threads accordingly.

B. *Parallel Nonlinear Interdependency Analysis with GPGPU*

This study focus on a GPGPU-aided implementation of the NLI application, referred to as the G-NLI approach. G-NLI has been designed to exploit the very fine-grained parallelisms at the NLI level. We have performed a preliminary study on the application's runtime performance, three basic operations contribute more than 99% of the execution time: (1) distance calculation (20%+), (2) sorting (50%+), and (3) computing NLI measures (20%+). Parallelization has been exactly performed to improve these operations (see Section III.*A*).

Given a 10-channel EEG dataset, the size of time window n set to 2048, embedding dimension m set to 8, time delay τ set to 4, the length of delay vectors is then set to N = (n - (m-1) τ) = 2020. After phase space reconstruction, we obtain 10 matrices corresponding to the 10 input epoches, each matrix contains 2010 delay vectors (8-dimension).

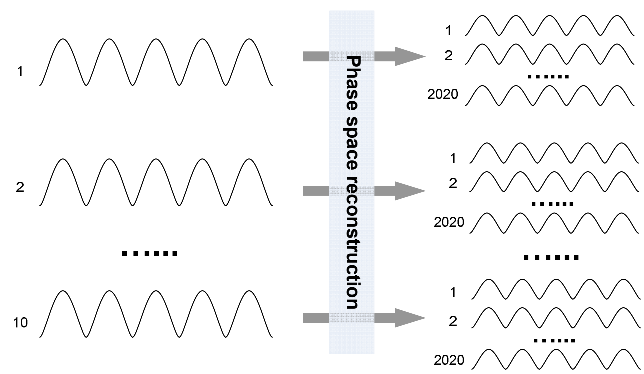

Figure 3. Phase space reconstruction for 10-channel EEG data.

The G-NLI enables parallel distance calculation. For each matrix (2020*8), G-NLI initiates 2020 CUDA blocks with each block maintaining 160 threads. Each block calculate $(\mathbf{x}_i\text{-}\mathbf{x}_j)^2$ for a designated delay vector against other 2020 vectors (including itself) in a completely parallel manner. NLI in total employs 2020*10 blocks and 2020*10*160 threads for processing the 10 matrices.

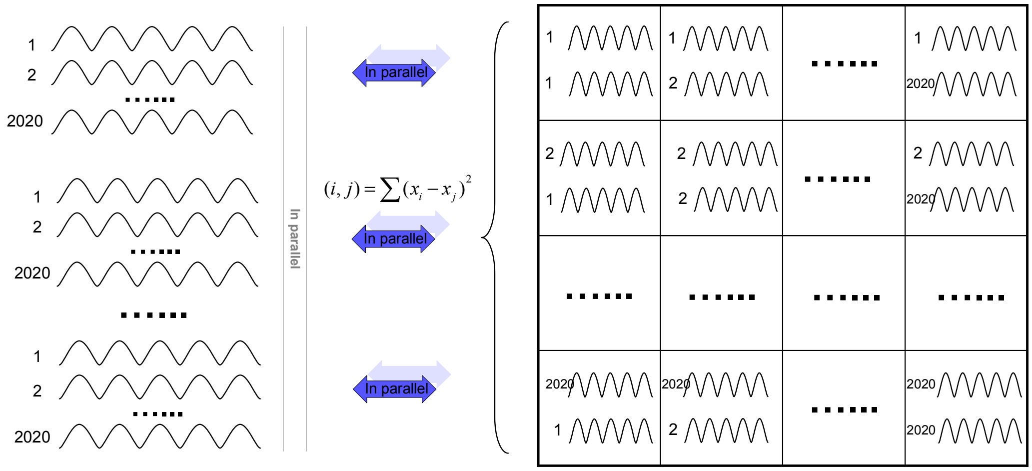

Figure 4. Parallel distance calculation with massive CUDA threads.

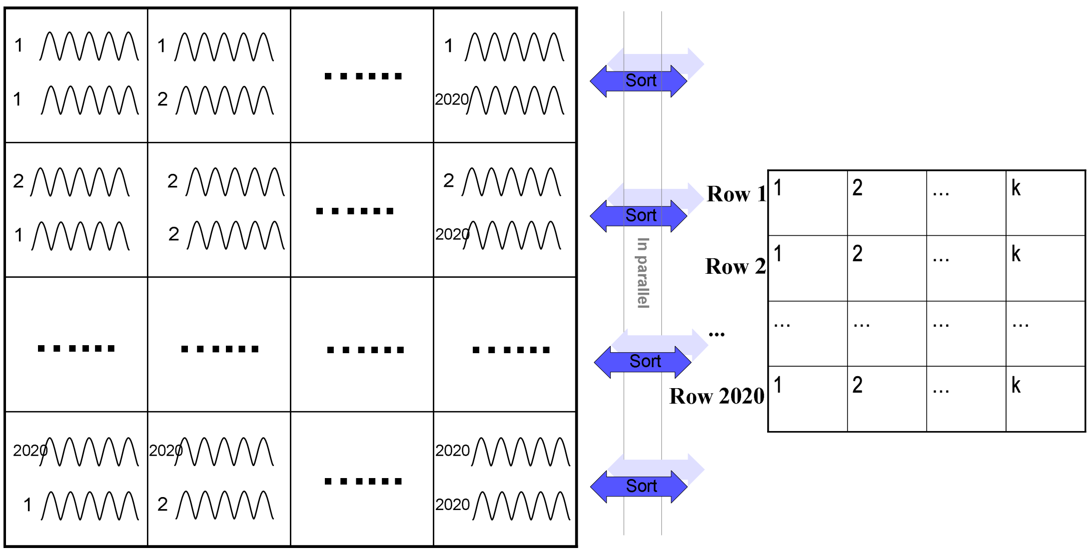

Figure 5. Parallel sorting with the bitonic sorting algorithm.

Sorting the elements of a large matrix (2020*2020) is an onerous task. G-NLI employs the bitonic sorting algorithm [8] to sort the distance matrix. Each row is filled in with 28 zeros to pad the row to have 2048 elements, thus the bitonic sorting algorithm can apply. G-NLI adopts a parallel implementation of the algorithm upon CUDA and obtains 10 large sorted matrices. The first k columns of each sorted matrix are retrieved as k nearest neighbors.

Calculation of Euclidean distance for each $\mathbf{x}_n$ is handled by an individual CUDA block thread, 2020 in total in the case presented in Figure 6. The squared mean Euclidean distance to its k closest neighbors are calculated using k concurrent threads. For calculating an $S^{(k)}(\mathbf{X}|\mathbf{Y})$, G-NLI assign 2020 threads for this task. Eventually, 2010*10 threads operate concurrently to solve the S measures for all 10 epochs. Same scheme applies for calculating H, N and M.

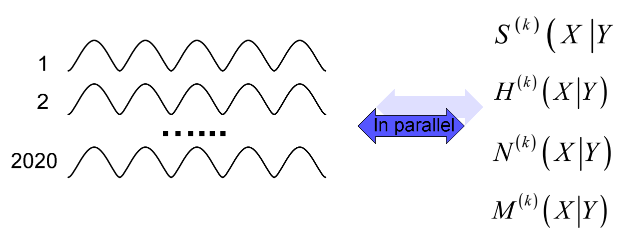

Figure 6. Parallel calculation of the NLI measures.

V. Experiments and Results

Experiments have been carried out to evaluate the performance of the G-NLI approach. We have also developed alternative NLI applications for comparison purpose, i.e., (1) the original serial NLI application, written as "Serial-NLI", and (2) another parallel NLI application ("PC-NLI") to be executed over CPU. We designed the PC-NLI application by parallelizing the Serial-NLI at the application level and implemented the approach in C++ using Windows multithreading technique. PC-NLI uses one CPU thread to encapsulate an individual NLI procedure.

Experiments have been conducted over a desktop computer equipped with a NVIDIA GeForce GTX 580 video card. The configurations of the execution environment are listed in Table I. The PC-NLI implementation has been executed over the desktop. The G-NLI approach has been implemented using NVIDIA CUDA 4.3. The G-NLI application is initiated from the desktop ("host") with most of its GPGPU-enabled computation executed over the video card ("device").

TABLE I. CONFIGURATIONS OF THE TESTBED

Host	**Table Column Head**		
	Operating System	***CPU***	***RAM***
Specifications	Windows 7 Pro 64	Intel I5 2300 （3.2GHz）	4G
Device	**NVIDIA GeForce GTX 580**		
	CUDA cores	***Processor Clock***	***Memory***
Specifications	512	1544M	1536 MB GDDR5

A. Materials

The EEG dataset (local field potential) was recorded from a patient suffering from temporal lobe epilepsy. The patient is a 20-year old male. The EEG dataset was collected using 6 electrodes and 4 electrodes placed on front and temporal regions respectively, with a sampling frequency 256Hz. The EEG dataset's length is 2000s. P1, P2, P3 and P4 highlight the interictal state, initial ictal state, overall ictal state and post-ictal state respectively.

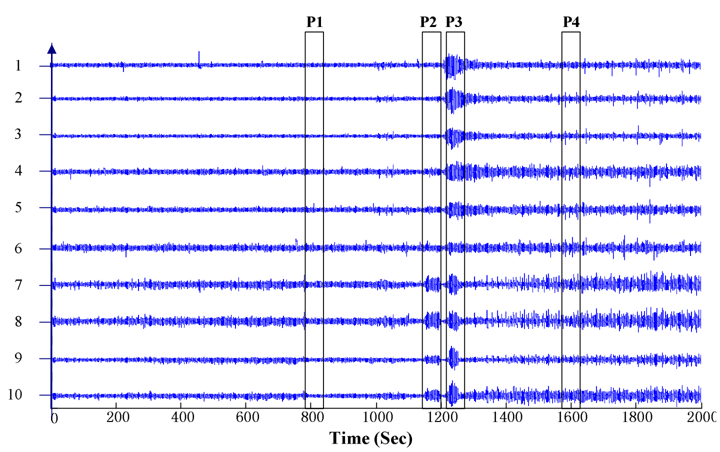

Figure 7. The 10-channel EEG dataset.

B. Runtime Performance

In the runtime performance test, we adopt the parameter settings as stated in Section IV.*B*. The EEG series has been divided into 499 time windows, i.e., 4990 epochs of time series.

The execution times of the three NLI implementations are about 833300s (Serial-NLI), 35150s (PC-NLI), and 531s (G-NLI). The execution time of Serial-NLI is prohibitively long and is very unsuited for practical applications. In contrast, PC-NLI gains speedup of 23 times. This dramatic performance improvement is gained by the CPU-based parallel computing technique.

Amazingly, G-NLI makes even a leap forward in terms of execution efficiency comparing to PC-NLI. It is even 65 times faster than PC-NLI does despite the latter runs on a cutting-edge high end desktop.

In addition to the dramatic performance improvement achieved by G-NLI, it should also be noted that the GPU used in this study is for home entertainment. Its cost is minimal in contrast to that of a high performance cluster. Notwithstanding, deployment of G-NLI and a GPU is very convenient. As G-NLI excels in execution efficiency, cost-effectiveness and practicability, it provides an ideal tool for real-time NLI analysis for EEG dataset.

C. Global NLI Measures

We define $Sum_i = S^{(k)}(\{\mathbf{X}\}|\mathbf{Y}) - S^{(k)}(\mathbf{Y}|\{\mathbf{X}\})$, which reflects the cumulative effect of **Y** (denoted by the *i*th channel) on the whole region under study. A negative value means **Y** is driven by the rest of the region, and a positive value means **Y** is driving the rest of the region. Figure 8 presents the results of the global *S* measures of the 10 channels.

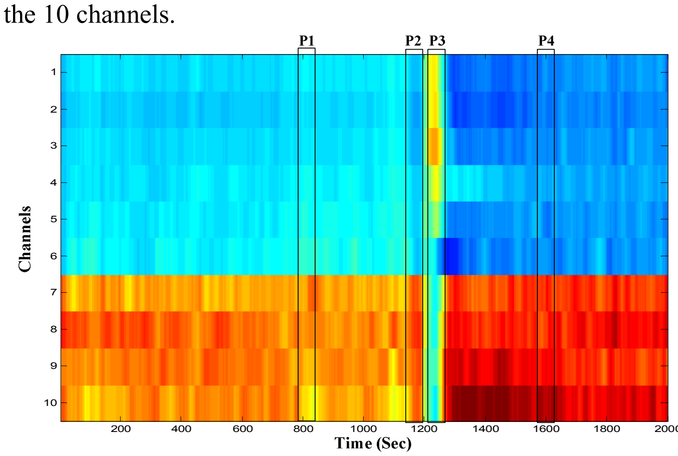

Figure 8. The Global NLI measure (*S*).

As indicated in Figure 8, in period P1, the 6 channels (1-6) at the front region have negative *Sum* values (blue), which imply the 6 channels are responses. On the contrary, the 4 channels (7-10) at the temporal region have positive *Sum* values (orange), which means these channels are driving the regions. In period P3, the driving force from the 4 channels reaches the maxim and intensively activates channels 1-6. The latter 6 channels exhibit a short period of positive values. These evidences lead to the conclusion that the region (channels 7-10) can be the epilepsy focus

VI. CONCLUSIONS AND FUTURE WORK

This study established a solution based on a many-core platform to satisfying the need of fast and scalable synchronization measurement with Nonlinear Interdependency analysis. We have developed a massively parallel NLI approach on top of CUDA. Our design utilizes the analysis application's built-in fine-grained parallelisms. The GPGPU-aided approaches naturally divide the application into a large number of fine-grained tasks and map the tasks to the same number of CUDA threads executed concurrently over possibly hundreds of GPU cores. The performance of our approach is inspiring as they are able to solve a problem in less than an hour while a serial NLI typically have to consume hundreds of hours. We achieved the success only using a commodity video card for home gaming.

We utilized the massively parallel NLI approach to analyze a multivariate EEG obtained from an epilepsy patient. We have derived the global synchronization measures of the EEG's channels. The measure values of different channels manifest an appropriate indicator of epilepsy focus. The results indicate that the proposed approach is promising in synchronization analysis on multivariate signals.

ACKNOWLEDGMENT

This work was supported in part by National Natural Science Foundation of China (grants No. 60804036 and 61102005), National Science Fund for Distinguished Young Scholars (61025019), the Hundred University Talent of Creative Research Excellence Programme (Hebei, China), the Programme of High-Resolution Earth Observing System (China), the Fundamental Research Funds for the Central Universities (CUGL100608, CUG, Wuhan), Specialized Research Fund for the Doctoral Program of Higher Education (20110145110010), and the Program for New Century Excellent Talents in University (grant No. NCET-11-0722).

Dr. Lizhe Wang's work in paper is funded by "One Hundred Talents" Programme of Chinese Academy of Sciences.

REFERENCES

[1] Brown R, Kocarev L. A unifying definition of synchronization for dynamical systems. Chaos 10: 344–349, 2000.

[2] Carmeli C, Knyazeva MG, Innocenti GM, De Feo O. Assessment of EEG synchronization based on state-space analysis. Neuroimage 25: 339–354, 2005.

[3] D. Chen, D. Li, M. Xiong, H. Bao, X. Li: GPGPU-Aided Ensemble Empirical Mode Decomposition for EEG Analysis during Anaesthesia, IEEE Transactions on Information Technology in BioMedicine, Vol. 14, Issue 6, pp 1417-1427, 2010.

[4] D. Chen, L. Wang, G. Ouyang, X. Li, "Massively Parallel Neural Signal Processing on a Many-Core Platform," IEEE Computing in Science and Engineering, vol. 13, no. 6, pp. 42-51, 2011.

[5] J. Fell, P. Klaver, K. Lehnertz, T. Grunwald, C. Schaller, C. E. Elger, and G. Fern´andez. Human memory formation is accompanied by rhinal-hippocampal coupling and decoupling. Nature Neurosci., 4:1259, 2001.

[6] Kindratenko V, Thiruvathukal, GK, Gottlieb S: High-Performance Computing Applications on Novel Architectures. Computing in Science and Engineering 2008; 10(6):13-15.

[7] Kraskov A. Dissertation (Ph.D. Thesis) NIC-Directors; Jülich: 2004. Synchronization and interdependence measures and their applications to the electroencephalogram of epilepsy patients and clustering of data.

[8] H. W. Lang. Bitonic sort. http://www.iti.fh-flensburg.de/lang/algorithmen/sortieren/bitonic/bitonicen.htm (Jan. 2012)

[9] Le Van Quyen M, Soss J, Navarro V, Robertson R, Chavez M, Baulac M, Martinerie J. Preictal state identification by synchronization changes in longterm intracranial EEG recordings. Clin Neurophysiol 116: 559–568, 2005.

[10] Xiaoli Li, Dong Cui, Premysl Jiruska, John E. Fox, Xin Yao, and John G. R. Jefferys. Synchronization Measurement of Multiple Neuronal Populations. J Neurophysiol 98:3341-3348, 2007.

[11] Mormann F, Lehnertz K, David P, Elger CE. Mean phase coherence as a measure for phase synchronization and its application to the EEG of epilepsy patients. Physica D 144: 358–369, 2000.

[12] Müller, A, Osterhage, H., Sowaa, R., Andrzejakc, R. G., Mormanna, F., Lehnertz, K.: A Distributed Computing System for Multivariate Time Series Analyses Of Multivariate Neurophysiological Data, Journal Of Neuroscience Methods 2006, 152(1-2):190-201.

[13] Rossant C, Goodman DFM, Fontaine B, Platkiewicz J, Magnusson AK and Brette R: Fitting neuron models to spike trains. Frontiers in Neurosciences 2011; 5:9. doi: 10.3389/fnins.2011.00009.

[14] F. Takens. Detecting strange attractors in turbulence. In D. A. Rand and L. S. Young, editors, Dynamical Systems and Turbulence, volume 898 of Lecture Notes in Mathematics, page 366. Springer-Verlag, Berlin, 1980.

[15] Vangelis Sakkalis, Ciprian Doru Giurcaneanu, Petros Xanthopoulos, Michalis E. Zervakis, Vassilis Tsiaras, Yinghua Yang, Eleni Karakonstantaki, and Sifis Micheloyannis. Assessment of linear and nonlinear synchronization measures for analyzing EEG in a mild epileptic paradigm. Trans. Info. Tech. Biomed. 13, 4 (July 2009), 433-441.

[16] F. J. Varela. Resonant cell assemblies, A new approach to cognitive functions and neuronal synchrony. Biol. Res., 28:81, 1995.

[17] Varela F, Lachaux JP, Rodriguez E, Martinerie J. The brainweb: phase synchronization and large scale integration. Nat Rev Neurosci 2: 229–239, 2001.

[18] J. A. Wilson and J. C. Williams, Massively Parallel Signal Processing Using the Graphics Processing Unit for Real-Time Brain-Computer Interface Feature Extraction, Front Neuroengineering, vol. 2, pp. 11, 2009.

[19] Lizhe Wang, Cheng Fu, Research Advances in Modern Cyberinfrastructure. New Generation Comput. 28(2): 111-112, 2010.

[20] Lizhe Wang, Marcel Kunze, Jie Tao, Gregor von Laszewski, Towards building a cloud for scientific applications. Advances in Engineering Software 42(9): 714-722, 2011.

[21] Lizhe Wang, Gregor von Laszewski, Andrew J. Younge, Xi He, Marcel Kunze, Jie Tao, Cheng Fu, Cloud Computing: a Perspective Study. New Generation Comput. 28(2): 137-146, 2010.

[22] Lizhe Wang, Jie Tao, Gregor von Laszewski, Holger Marten, Multicores in Cloud Computing: Research Challenges for Applications. JCP 5(6): 958-964, 2010.