2014 International Symposium on Nonlinear Theory and its Applications
NOLTA2014, Luzern, Switzerland, September 14-18, 2014

Fast Recurrence Quantification Analysis on GPUs

Tobias Rawald[1,2], Mike Sips[2], Norbert Marwan[3], and Doris Dransch[1,2]

[1] Humboldt University of Berlin
[2] Helmholtz Centre Potsdam - GFZ German Research Centre for Geosciences
[3] Potsdam Institute for Climate Impact Research
Email: (trawald, sips, dransch)@gfz-potsdam.de, marwan@pik-potsdam.de

Abstract—In this paper we present a novel computing approach for conducting recurrence quantification analysis. It utilises the concepts of *Divide and Recombine* to distribute the detection of vertical and diagonal lines in a recurrence matrix among multiple graphics card processors. We employ our approach to calculate recurrence quantification measures based on an hourly air temperature record of Potsdam, a time series with more than one million data points. The proposed calculation scheme reduces the computing time drastically; from more than 6 h using a single-threaded CPU implementation to about 5 min using two graphics card processors.

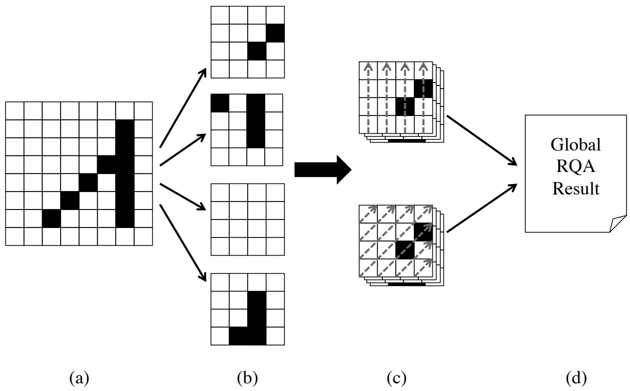

Figure 1: Our computing approach.

1. Introduction

Recurrence plots (RPs) and recurrence quantification analysis (RQA) are powerful instruments for analysing recurrences in measured time series [1]. Their application in many fields have proven their potential for various kinds of analyses [2, 3, 4, 5, 6]. Small scale structures in a RP, such as diagonal lines, are used to define measures of complexity, establishing the RQA [1, 7, 8].

The time complexity of calculating the RQA measures is $O(N^2)$, prohibiting an efficient computation for very long time series. Furthermore, current implementations of RQA are restricted concerning memory usage. The *Cross Recurrence Plot* Toolbox for MATLAB® [9] is limited to $N < 10,000$ data points when calculating the entire RP. Likewise, the RQA software by Webber Jr. [10] is capable of processing only up to $N = 5,000$ data points.

Investigating the existing implementations, we developed a computing approach that builds on the concepts of *Divide and Recombine* (D&R) [11]. We illustrate the capabilities of our approach in comparison to existing implementations using an application example from climate impact research.

2. Our Approach

2.1. Divide and Recombine

Divide and Recombine is a general approach to address large computational problems. The basic idea is to divide a data set into small sub sets allowing the fast computation of analytical results of the sub sets. The intermediate results of the sub sets are recombined into a global solution.

We apply D&R to recurrence quantification analysis as follows. The underlying recurrence matrix of a RP is divided into multiple sub matrices (see Fig. 1(a), Fig. 1(b)). Within each sub matrix, we detect all vertical and diagonal lines, storing this information in local histograms. The key contribution of our approach is that the detection of lines within a single sub matrix is performed by a graphics card processor. Its architecture allows to execute a large number of vertical and diagonal line detection tasks simultaneously (depicted as dotted arrows in Fig. 1(c)). The local histograms are recombined into two global histograms; one for vertical and one for diagonal lines. The recombination process is straightforward: We add up the local histograms, either referring to vertical or diagonal lines, to a global histogram. Based on the two global histograms, we compute the global RQA measures (see Fig. 1(d)).

2.2. Detection of Vertical and Diagonal Lines

An important challenge in the context of RQA is that vertical and diagonal lines may spread over multiple sub matrices. This is the reason why Divide and Recombine can not be used as a wrapper for existing RQA implementations. This approach would compute only valid results for the individual sub matrices. Enforcing the computation of valid global frequency distributions of vertical and diagonal lines, we introduce *carryover buffers*; one for vertical and one for diagonal lines. The basic purpose of a carryover buffer is to share information about the length of vertical or diagonal lines that exceed the borders of a sub matrix.

For each column, the vertical carryover buffer stores the length of all vertical lines that exceed the horizontal border of a sub matrix. If a vertical line reaches the last element of a column of a sub matrix, the corresponding carryover buffer element stores its current length. This value is used as input for determining vertical lines in the vertically adjacent sub matrix. As a precondition, the sub matrices have to be processed in a specific order (see Fig. 2).

This concept can easily be adapted to the detection of diagonal lines, including the use of a carryover buffer and a particular order of processing concerning the set of sub matrices. The major difference is that a diagonal line may transcend not only the horizontal but also the vertical borders of sub matrices. Furthermore, the size of the carryover buffer is equivalent to the number of diagonals (not the number of diagonal lines) within the original recurrence matrix. Figure 3 depicts the detection of diagonal lines. To illustrate, the RP in this figure contains only a single diagonal line of length 5.

2.3. Discussion of Parameters

As a precondition, we subdivide the recurrence matrix into quadratic sub matrices with a predefined edge length; except the sub matrices at the borders of the original recurrence matrix. In general, the user can choose an arbitrary edge length. Nevertheless, experiments have shown that the size of the sub matrices highly influences the overall computing time. Therefore, it should adhere to the specifics of the GPU architecture as well as the programming framework applied. Future work is to investigate the impact of sub matrix size to the runtime in detail.

In addition, not all sub matrices share information about the same diagonals and verticals of the original recurrence matrix. Therefore, they do not have to exchange information using the carryover buffer. Generally, sub matrices which do not share any element of the carryover buffer can be processed concurrently. Our approach allows to utilise multiple GPU devices by processing a number of sub matrices at the same time. This reduces the overall computing time additionally.

3. Application to Climate Data

We use the hourly air temperature dynamics in Potsdam, referred to as *Potsdamer Klimareihe*, to demonstrate the benefits of our approach. We consider the period from 1893 until 2011, resulting in 1,043,112 data points.

We analysed the the *Potsdamer Klimareihe* using our D&R approach. We provide an implementation based on the OpenCL™ framework for parallel programming of heterogeneous systems [12]. The hardware setup of the experiment consists of a standard desktop workstation, con-

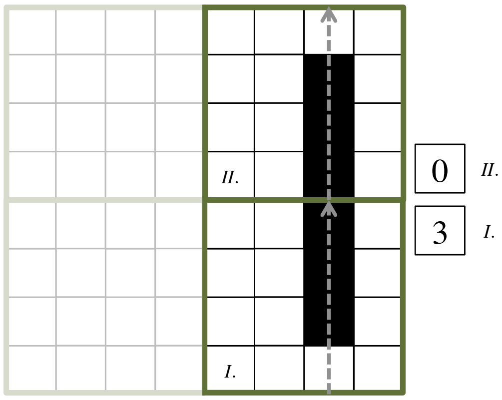

Figure 2: Detection of vertical lines. To compute a valid global RQA result, the sub matrices have to be processed in ascending order (*I.* → *II.*). This is due to the reason that the column elements of the original recurrence matrix have to be processed in ascending order. The corresponding carryover buffer element stores the lengths (3 and 0) of the vertical lines detected (if present) at the horizontal borders of *I.* and *II.* The intermediate states of the carryover buffer element after processing each sub matrix are depicted on the right.

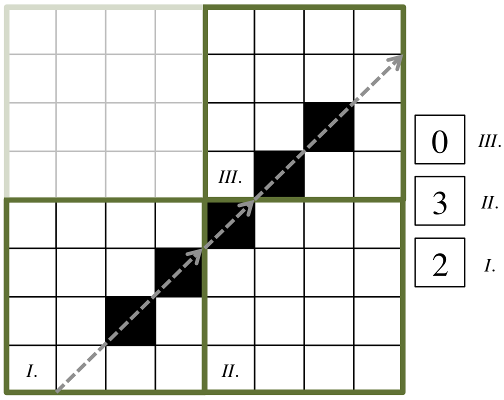

Figure 3: Detection of diagonal lines. Similar to the detection of vertical lines, the sub matrices have to be processed in ascending order (*I.* → *II.* → *III.*), referring to the ascending order of the diagonal elements within the original recurrence matrix. The corresponding carryover buffer element stores the lengths (2, 3 and 0) of the diagonal lines detected (if present) at the horizontal and vertical borders of *I.*, *II.* and *III.* The intermediate states of the carryover buffer element after processing each sub matrix are depicted on the right.

taining an Intel® Core™ i5-3570 quad-core CPU at up to 3.80GHz and 16GB of main memory. It also includes a NVIDIA® GeForce® GTX™ 690 that provides two graphics card processors running at up to 1.019GHz; each of them is supplied with 2GB of memory. In the context of heterogeneous computing, each graphics card processor is treated as a separate computing device. The workstation runs on a 64-bit version of *OpenSuse* 12.1 with version 1.1 of OpenCL™.

To demonstrate its scale-out capabilities, we compare our massively parallel GPU implementation to two non-D&R CPU implementations (see Tab. 1, Tab. 2). The single-threaded RQA implementation refers to version 1.13z of the *Commandline Recurrence Plots* tool, which is available at `http://tocsy.pik-potsdam.de/commandline-rp.php`. The multi-threaded is an extension of the single-threaded implementation running multiple CPU threads. Our approach allows the utilization of a single GPU device (*1x*) or both GPU devices (*2x*) available in our computing environment.

Technology	Runtime
CPU - Single Thread	6 h 18 min 4 sec
CPU - Multiple Threads	1 h 33 min 58 sec
GPU - 1x	10 min 11 sec
GPU - 2x	5 min 10 sec

Table 1: Runtimes for RQA calculations for the full time series of hourly temperature anomaly data of Potsdam from 1893 to 2011.

Technology	Runtime Improvement
CPU - Single Thread	≈ 73 times
CPU - Multiple Threads	≈ 18 times
GPU - 1x	≈ 2 times

Table 2: Runtime improvements considering the OpenCL™ implementation running on two GPU devices as a reference; based on the runtime measurements presented in Tab. 1.

The runtime experiment shows that our D&R approach calculates the RQA measures in roughly five minutes (see Tab. 1, Fig. 4). Note, the RQA computations employing the *Commandline Recurrence Plots* tool running on a CPU takes over six hours. This renders a comprehensive analysis of this data set impossible. In contrast, the fast computation of the RQA measures using our D&R approach enables a comprehensive analysis of the Potsdam temperature profile. It allows to calculate RQA measures for a variety of combinations of embedding dimension m, time delay τ and similarity threshold ϵ in a reasonable amount of time.

We observed the four RQA measures *recurrence rate* (*RR*), *determinism* (*DET*), *average diagonal line length* (*L*), and *laminartity* (*LAM*) [1] from the seasonally corrected *Potsdamer Klimareihe* (anomaly values); using a recurrence threshold of $\varepsilon = 1$ (Euclidean norm). These measures reflect different aspects of the short-term dynamics; e.g., predictability.

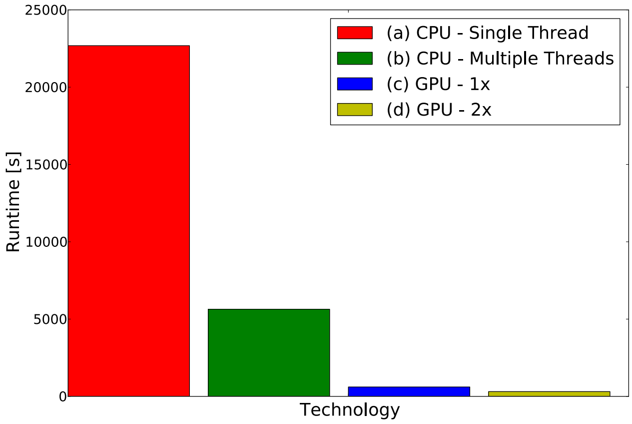

Figure 4: Runtime comparison. Runtimes of the RQA computations for the full time series of hourly temperature anomaly data of Potsdam from 1893 to 2011 using different implementations.

Our computational approach provides the following initial observations. We find that all four RQA measures do not remarkably change for the full period and the sub periods 1893–1974 and 1975–2011 (see Tab. 3). This result suggests that, in contrast to the longer time-scales, the short-term dynamics, including the short-term weather predictability, has not (yet) changed due to the climate change. For the period between 1893 and 1974, the warming trend of the annual mean temperature was 0.46 K per century, but after 1974 the trend rose to 3.4 K per century (see Fig. 5).

To study the short-term dynamics, we remove the annual trend (seasonal cycle) from the data by phase averaging, resulting in an anomaly temperature record. We use a time delay embedding of dimension $m = 5$ and delay $\tau = 3$, which have been found by false nearest neighbors approach for finding m [13] as well as a combined autocorrelation and visual recurrence plot inspection approach for finding an optimal τ [14].

Table 3: RQA results for the full time series of hourly temperature anomaly data of Potsdam as well for the two periods 1893–1974 and 1975–2011.

Measure	1893–2011	1893–1974	1975–2011
RR	0.12	0.12	0.13
DET	0.94	0.94	0.94
L	8.4	8.4	8.6
LAM	0.96	0.97	0.96

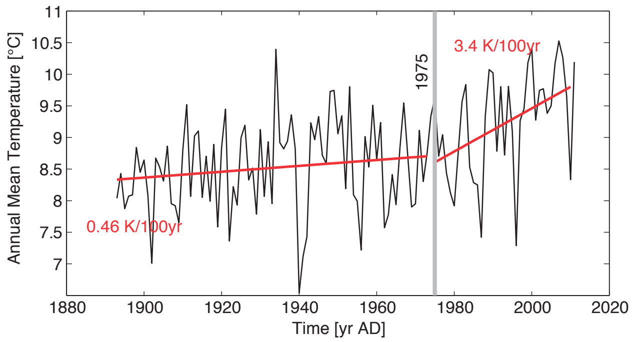

Figure 5: Potsdamer Klimareihe. Warming trend for the periods 1893–1974 and 1975–2011.

4. Conclusion

We present a novel computing approach for conducting RQA that combines the concepts of D&R with the parallel computing capabilities of GPUs. Our approach allows to compute RQA measures for the full *Potsdamer Klimareihe* in close to 5 min. A particular challenge is to compute valid RQA results. Therefore, it is required to share information between sub matrices, which we solve by introducing carryover buffers.

Our approach offers the following benefits:

- The carryover buffers and the enforcement of specific processing orders allow to reduce the overall computing time by utilising multiple GPU devices at the same time.
- The architectural design of GPU devices enables the massively parallel detection of vertical and diagonal lines within a single sub matrix.
- The usage of the OpenCL$^{\mathrm{TM}}$ framework allows running the implementation on a variety of different hardware architectures.
- An implementation of our approach and further information will soon be available at `http://www.gfz-potsdam.de/fast-rqa`.

Acknowledgements

We would like to thank T. Nocke and F.-W. Gerstengarbe for fruitful discussions and acknowledge support from the Potsdam Research Cluster for Georisk Analysis, Environmental Change and Sustainability (PROGRESS, support code 03IS2191B).

References

[1] Marwan N, Romano MC, Thiel M, Kurths J. Recurrence Plots for the Analysis of Complex Systems. Physics Reports. 2007;438(5–6):237–329.

[2] Marwan N. A Historical Review of Recurrence Plots. European Physical Journal – Special Topics. 2008;164(1):3–12.

[3] Donges JF, Donner RV, Trauth MH, Marwan N, Schellnhuber HJ, Kurths J. Nonlinear detection of paleoclimate-variability transitions possibly related to human evolution. Proceedings of the National Academy of Sciences. 2011;108(51):20422–20427.

[4] Goswami B, Marwan N, Feulner G, Kurths J. How do global temperature drivers influence each other? – A network perspective using recurrences. European Physical Journal – Special Topics. 2013;222:861–873.

[5] Ponyavin DI. Solar Cycle Signal in Geomagnetic Activity and Climate. Solar Physics. 2005;224(1–2):465–471.

[6] Muller RA, MacDonald GJ. Ice Ages and Astronomical Causes (Springer Praxis Books / Environmental Sciences). Springer; 2002.

[7] Zbilut JP, Webber Jr CL. Embeddings and delays as derived from quantification of recurrence plots. Physics Letters A. 1992;171(3–4):199–203.

[8] Webber Jr CL, Zbilut JP. Dynamical assessment of physiological systems and states using recurrence plot strategies. Journal of Applied Physiology. 1994;76(2):965–973.

[9] Marwan N. CRP Toolbox 5.17; 2013. Platform independent (for Matlab). Available from: `http://tocsy.pik-potsdam.de/CRPtoolbox`.

[10] Webber Jr CL. RQA Software 14.1; 2013. Only for DOS. Available from: `http://homepages.luc.edu/~cwebber`.

[11] Guha S, Hafen R, Rounds J, Xia J, Li J, Xi B, et al. Large complex data: divide and recombine (D&R) with RHIPE. Stat. 2012;1(1):53–67.

[12] OpenCL 1.1 Specification; 2010. Available from: `http://www.khronos.org/registry/cl/specs/opencl-1.1.pdf`.

[13] Kennel MB, Brown R, Abarbanel HDI. Determining embedding dimension for phase-space reconstruction using a geometrical construction. Physical Review A. 1992;45(6):3403–3411.

[14] Marwan N. How to avoid potential pitfalls in recurrence plot based data analysis. International Journal of Bifurcation and Chaos. 2011;21(4):1003–1017.