Dariusz PUCHALA, Bartłomiej SZCZEPANIAK, Mykhaylo YATSYMIRSKYY

Lodz University of Technology

doi:10.15199/48.2015.07.17

Lattice structure for parallel calculation of orthogonal wavelet transform on GPUs with CUDA architecture

Abstract. *In this paper, we present a novel approach to calculation of discrete wavelet transform (DWT) on modern Graphics Processing Units (GPUs) with CUDA architecture which takes advantage of highly parallel lattice structure. The experimental results obtained for model signals show that the proposed approach allows to obtain several times acceleration compared with sequential calculations carried out on the CPU while taking into account not only time of calculations but also time required for data transfers.*

Streszczenie. *W artykule zaproponowano nowe podejście do obliczania dyskretnego przekształcenia falkowego (DWT) na nowoczesnych procesorach graficznych (GPUs) o architekturze CUDA oparte o wysoce równoległe struktury kratowe. Wyniki badań eksperymentalnych przeprowadzonych na sygnałach modelowych pokazują, że zaproponowane podejście daje możliwość uzyskania kilkukrotnego przyśpieszenia obliczeń w porównaniu do obliczeń sekwencyjnych na CPU, biorąc pod uwagę nie tylko czas obliczeń, ale również czasy przesyłu danych.*
(Równoległe obliczanie ortogonalnych przekształceń falkowych na procesorach GPU o architekturze CUDA w oparciu o struktury kratowe)

Keywords: discrete wavelet transform, lattice structures, GPU computations, CUDA architecture
Słowa kluczowe: dyskretne przekształcenie falkowe, struktury kratowe, obliczenia na GPU, architektura CUDA

Introduction

The discrete wavelet transform (DWT) is widely used in many practical applications such as signal processing and analysis, data compression [1], clustering or data mining [2, 3], etc. It should be noted, however, that the constant growth of the capabilities of modern data acquisition devices causes the continuous increase in the sizes of datasets to be processed. Hence, it is still a very actual task to search for more computationally efficient realizations of DWT which are suitable for processing of one- as well as multidimensional data. In previous years, we could observe the development of new computational techniques for calculation of DWT that take advantage of lifting scheme [4] or lattice structures [5, 6]. Both mentioned approaches are highly parallel and computationally efficient. However, the lattice structure based approach is highly unified and hence, we choose it as a candidate for implementation on modern GPU devices.

In this paper, we propose a parallel approach to calculation of orthogonal DWT on modern GPUs with CUDA architecture that takes advantage of lattice structure proposed in paper [5]. It is proved experimentally that the proposed solution is several times faster than the sequential one while taking into account not only the time of calculations but also the time required by data transfer in both directions between host computer and GPU card. In the last section of this paper, we determine the future steps that should be performed in order to improve the presented approach.

Lattice structure for calculation of DWT

There is a lot of approaches enabling to describe and calculate discrete wavelet transform. The basic approach is a filter bank. It is a one input and two outputs systems representing filtering operations with low-pass filter h and high-pass filter g (see Fig. 1).

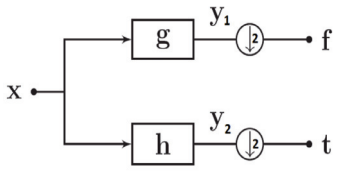

Fig. 1. Filter bank model for calculation of DWT

As a result of such filtering, we obtain two signals having the same length as the input signal. For that reason after performing calculations in filter bank, it is necessary to apply the decimation process. In digital signal processing, decimation is the process of reducing the sampling rate of a signal. The decimation factor is usually an integer value greater than one (in DWT it is usually 2).

Wavelet synthesis method presented in this paper is based on lattice structure - a mathematical schema which may represent various wavelet transforms [5]. The basis for this structure is a two-point base operation D_l defined as a 2 on 2 element matrix of the form (1). Then the output of a two-point base-operation y_1 and y_2 is calculated on the basis of two input values x_1, x_2 and proper factor values a_l, b_l, c_l and d_l as $y_1 = a_l x_1 + b_l x_2$ and $y_2 = c_l x_1 + d_l x_2$.

$$D_l = \begin{bmatrix} a_l & b_l \\ c_l & d_l \end{bmatrix}. \tag{1}$$

The forward lattice structure is composed of $K/2$ stages numbered by $l = 1, 2, \ldots, K/2$ (where K is a natural even number), each containing D_l operations repeated $N/2$ times, where K and N are the lengths of the filter's impulse response and of a processed signal respectively. On each stage of the lattice structure pairs of signal elements are calculated by D_k base operations. Each pair of the consecutive stages are connected with each other by a cyclic N-point downward shift. That means that the lower input of the last base operation in the current stage is connected to the upper output of the first base operation in the preceding stage. The same approach may be used to calculate the inverse lattice structure which is in fact the reversed forward structure where the base operation D_l is substituted by inverse operation ${D_l}^{-1}$ and the mentioned shift after each stage is performed in the upper direction. In order to calculate $K/2$ stages while operating on N sample signals it is necessary to perform $(N/2)(K/2)$ base operations D_l each with two additions and four multiplications.

The described lattice structure is used to calculate discrete wavelet transform (DWT). Upper outputs of base operations in last layer will be referred to as the "low-pass outputs", and lower outputs will be referred to as the "high-pass outputs". The main problem in parallel implementation of such algorithm is the mentioned shift of base operations after each stage.

Factorization of orthogonal DWT into lattice structure

Two-channel filter bank which fulfills the orthogonality condition (perfect reconstruction condition) can be factorized into a forward lattice structure with use of scheme depicted in Fig. 2. In order to achieve it, it is necessary to use symmetrical C_l and asymmetrical S_l base operations, defined as

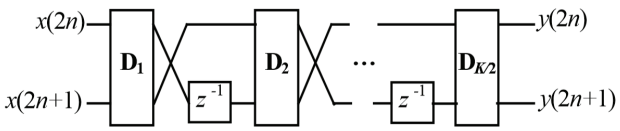

Fig. 2. Building scheme of forward lattice structure

follows:

$$C_l = C_l^{-1} = \begin{bmatrix} a_l & b_l \\ b_l & -a_l \end{bmatrix}$$

for $l = 0, 1, \ldots, K/2 - 1$. The determinant of C_l matrix is denoted as $\det(C_l)$ and equals:

$$\det(C_l) = -{a_l}^2 - {b_l}^2 = -1.$$

In the second case forward base operation S_l and the inverse base operation ${S_l}^{-1}$ are defined as follows:

$$S_l = \begin{bmatrix} a_l & b_l \\ -b_l & a_l \end{bmatrix}, \; S_l^{-1} = \begin{bmatrix} a_l & -b_l \\ b_l & a_l \end{bmatrix}.$$

Both presented orthogonal base operations may be chosen for filter bank factorization. Next step in factorization algorithm relies on finding the recursive dependence between impulse responses for filter pairs H_M, G_M and H_{M-2}, G_{M-2}, where $M = K, K-2, ..., 4$. This recursive dependence obtained directly from the form of the forward lattice structure is presented below:

$$\begin{bmatrix} h_{M,M-1} & h_{M,M-2} & \cdots & h_{M,1} & h_{M,0} \\ g_{M,M-1} & g_{M,M-2} & \cdots & g_{M,1} & g_{M,0} \end{bmatrix} = C_{M/2}$$

$$\begin{bmatrix} g_{M-2,M-3} & g_{M-2,M-4} & g_{M-2,M-5} & \cdots & g_{M-2,0} & 0 & 0 \\ 0 & 0 & h_{M-2,M-3} & \cdots & h_{M-2,2} & h_{M-2,1} & h_{M-2,0} \end{bmatrix}.$$

Now finding the elements of filters H_{M-2}, G_{M-2} and base operation $C_{M/2}$ is necessary. Performing few mathematical operations, including multiplications, reducing by two elements the length of considered filters and continuing recursively this process of reduction of the lengths of considered filters for $M = K, K-2, ..., 4$ may finally result in obtaining the base operation C_k for each of the stages, except for the first stage of the considered factorization. In order to obtain the elements of the base operation in first stage it is necessary to use the equality which holds for D_1

$$\begin{bmatrix} c_1 & d_1 & 0 & 0 \\ 0 & 0 & a_1 & b_1 \end{bmatrix} = \begin{bmatrix} g_{2,1} & g_{2,0} & 0 & 0 \\ 0 & 0 & h_{2,1} & h_{2,0} \end{bmatrix}$$

what completes the factorization.

Implementation of lattice structure

In the experimental part of this paper, we concentrate on the case of two-dimensional data (e.g. images). However, taking into account an assumption of the separability of DWT, which is usually taken in practice of digital signal processing, it is possible to calculate DWT in the cases of any dimensionality of input data using the well known row and column approach. It should be also noted that such approach requires only the effective implementation of one-dimensional DWT. Moreover, for the purpose of calculation of DWT for more than one level of data filtering, we use the Mallat's scheme [7] which consists in applying at each level both low-pass and high-pass filters to the output of low-pass filter calculated at the preceding level.

It should be noted that the lattice structure described in previous sections is highly parallel. It is understood that at each stage all base operations can be calculated in a parallel mode. However, the write-before-read dependence between consecutive stages requires additional synchronization. The other issue is the mentioned cyclic shift of data between stages.

In the proposed implementation of one-dimensional DWT for CUDA architecture, we assign an arbitrary number of threads (typically 32 threads) to one input vector. These threads are responsible for calculation of outputs of base operations D_l in the following stages of lattice structure. Each thread handles some number of base operations D_l resulting from the uniform assignment of $N/2$ operators to available threads. Since threads operating on the same input vector are executed in one block, it is simple to synchronize their work between consecutive stages using standard __syncthreads() function. In order to solve the data shift issue, we introduce an additional memory buffer of the same size N as input vector, which allows for full flexibility of data handling between stages. Then it is enough to decrement indexes of outputs of operators D_l at each stage. Moreover, the additional buffer allows for convenient allocation of outputs of the last stage where the following order is desired: low-pass and high-pass outputs should be placed in the lower and upper parts of the buffer respectively.

In case of GPUs calculations the best performance can be achieved while executing a lot of arithmetic operations on the same dataset. Only then the acceleration resulting from parallel calculations can compensate time required by data transfer between host computer and GPU device. Hence, in the proposed approach, we assume that implemented one-dimensional DWT operates not on a single but on a big number of input vectors. Then the following vectors are processed inside different blocks. If the number of available blocks is not high enough then vectors are grouped into sets of appropriate sizes and then they are processed with use of separate kernel calls.

Experimental results

For experimental verification of the suitability of lattice structure for calculating DWT on GPUs with CUDA architecture several experiments involving two-dimensional model signals were performed. The case of two-dimensional data is very popular in practical applications since it concerns the tasks of image processing and analysis. However, as it was mentioned in previous section of this paper, the proposed implementation of DWT can be used to operate on data with any dimensionality.

For the sake of comparison the conditions under which we performed the experiments were established in the way that would be most similar to the conditions described in paper [8]. The authors of the mentioned paper proposed CUDA based implementation of two-dimensional DWT (only Haar wavelets) with application to data clustering. Hence, we used datasets composed of two-dimensional randomized square images containing the particular number of black, round spherical clouds of random points. Exemplary results in the form of plots presenting times of calculations performed on host computer (CPU) and GPU are shown in Figures 3 and 4. The experiments were carried out for several values of N (where N describes the size of image with N on N pixels), four levels of filtering and two different orders of filters $6/6$ (see Fig. 3) and $10/10$ (see Fig. 4).

Moreover, all experiments were conducted on a Windows laptop computer equipped with NVIDIA GT720M 2GB RAM Graphics Processing Unit and Intel i5 Central Process-

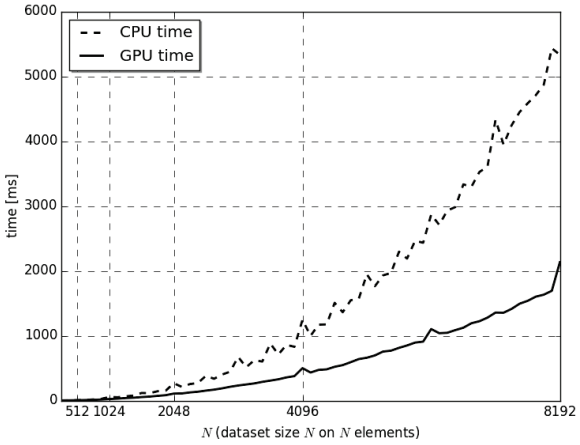

Fig. 3. Experimental results obtained for two-dimensional signals with four levels of filtering and filters of order $6/6$

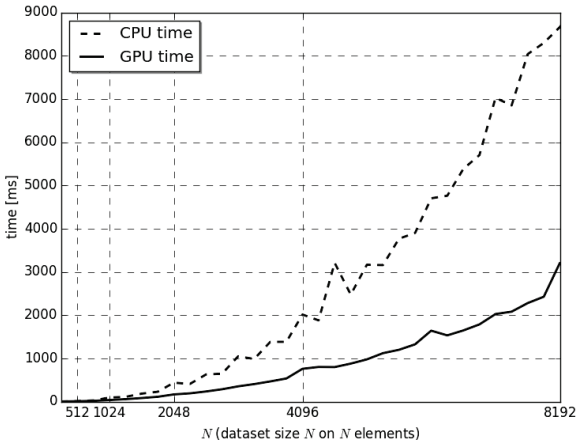

Fig. 4. Experimental results obtained for two-dimensional signals with four levels of filtering and filters of order $10/10$

ing Unit (2.60 GHz, 8 GB RAM).

Table 1. Times t_{tr} required for data transfer in both directions between host computer and GPU device (for two-dimensional datasets with sizes N on N elements)

N	512	1024	2048	4096	8192
t_{tr} [ms]	1.41632	4.51503	16.38880	64.7688	256.102

It should be noted that time of calculations is not the only criterion that should be taken into account while comparing the effectiveness of algorithms implemented on GPU and CPU. In order to perform calculations on GPUs, we have to send variables between host computer and device memories which also requires a significant amount of time. Table 1 presents exemplary amounts of time needed to send variables between CPU and GPUs.

After an analysis of results depicted in Figures 3 and 4, we may come to an undisputed conclusion that parallel implementation of discrete wavelet transform on GPU device with aid of lattice structure is more efficient and faster than calculations performed on host computer using the same structure. This advantage is more visible for large datasets. For example with dataset size 512 on 512 points GPU is at about 1.3 times faster. But when the size grows up to 8192 on 8192 points the speed up reaches the level of 2.71. The peak observed value was at level of 4.

The results presented in Table 1 allow to estimate the contribution of data transfer time. In the performed experiments it was at the level smaller than 10% of the total measured times.

It is also worth noting that the experiments where performed on a typical low budget device, not the most powerful GPU card available. In future scientific work, the authors plan to use one of the most powerful GPUs compatible with CUDA architecture in order to check the possible gain in time of execution.

Conclusion and directions of future research

It can be seen from the obtained results (c.f. Fig. 3, 4) that the proposed approach to parallel calculation of DWT for GPU devices with CUDA architecture allows to obtain significant (almost threefold) acceleration of calculations in comparison to calculations performed on host computer in a sequential way. Hence, we can state that the lattice structure based DWT algorithm is suitable for parallel realizations on parallel architectures.

Moreover, we can state that further development of massively parallel algorithms for solving various data processing problems will undoubtedly have a significant influence on such scientific areas as computer vision and recognition, medical ultrasonography, computer tomography or data compression, etc.

REFERENCES

[1] Colderbank A. R., Daubechies I., Sweldens W., Boon-Lock Yeo: Lossless Image Compression Using Integer to Integer Wavelet Transform, Proceedings International Conference On Image Processing, Vol. 1, 1997.

[2] Li T., Li Q., Zhu S., Ogihara M.: A survey on Wavelet Applications in Data Mining, SIGKDD Explorations, Vol. 4, Issue 2, 2002.

[3] Dash P. K., Iian Lee Wen Chun, Chilukuri M. V.: Power Quality Data Mining Using Soft Computing And Wavelet Transform, TENCON 2003.Conference on Convergent Technologies for the Asia-Pacific Region, Bangalore, India, 2003.

[4] Uytterhoeven G., Rose D., Bultheel A.: Wavelet Transform Using the Lifting Scheme, Report ITA-Wavelets-WP1.1, 28 April 1997.

[5] Yatsymirskyy M.: Lattice structures for synthesis and implementation of wavelet transforms, Journal of Applied Computer Science, 17(1), pp. 133–141, 2009.

[6] Yatsymirskyy M., Stokfiszewski K.: Effectiveness of lattice factorization of two-channel orthogonal filter banks, Joint Conference NTAV/SPA 2012, September 2012.

[7] Mallat S.: A theory for multiresolution signal decomposition: the wavelet representation, IEEE Transactions on Pattern Analysis and Machine Intelligence, 11(7), pp. 674-–693, June 1989.

[8] Artu Yildrim A., Ozdogan C.: Parallel wavelet-based clustering algorithm on GPUs using CUDA, Procedia Computer Science, 3 (2011), pp. 396–400, 2011.

***Authors**: Prof. Mykhaylo Yatsymirskyy, Ph.D. Dariusz Puchala, Institute of Information Technology, Lodz Technical University, ul. Wólczańska 215, 90-924 Lodz, Poland, email: mykhaylo.yatsymirskyy@p.lodz.pl, email: dariusz.puchala@p.lodz.pl, M. Sc. Bartłomiej Szczepaniak, Institute of Applied Computer Science, Lodz Technical University, ul. Stefanowskiego 18/22, Lodz, Poland, email: bartlomiej.szczepaniak@p.lodz.pl*