Procedia of Engineering and Medical Sciences

Proceedings of the International Congress on "Medical Improvement and Natural Sciences" | 2022

Processor Architectures in Data Base Problems

Rakhimov Bakhtiyar Saidovich [1]
Bekchanov Bakhtiyar Gafurovich [2]
Jumaniyazova Tupajon Alimovna [3]
Saidov Atabek Bakhtiyarovich [4]

[1] *Head of the Department of Biophysics and information technologies of Urgench branch of Tashkent Medical Academy, Uzbekistan*
[2] *Head of the Department of Children's propaedeutics*
[3] *Senior teacher of the Department of Biophysics and information technologies of Urgench branch of Tashkent Medical Academy, Uzbekistan*
[4] *student 4 – course Urgench branch of Tashkent University of Information Technologies named after Muhammad al Khwarizmi, Uzbekistan*

Annotation. Computer vision as a scientific discipline refers to the theories and technologies for creating medical database systems that receive information from an image. Despite the fact that this discipline is quite young, its results have penetrated almost all areas of life. Computer vision is closely related to other practical areas like image processing, the input of which is two-dimensional images obtained from a camera or artificially created. This form of image transformation is aimed at noise suppression, filtering, color correction and image analysis, which allows you to directly obtain specific information from the processed image. This information may include searching for objects, feature points, segments, etc. All scalar processors operate in SIMD mode, while executing a block of threads in the G80, the number of threads in a block is 32, called a warp. At the same time, in 4 clock cycles of the multiprocessor, all beam streams are processed at once when performing operations with floating point, with double precision - in 32 clock cycles, and transcendental functions - in 16 clock cycles. The number of threads per multiprocessor is limited. To synchronize the threads, special instructions have been developed that interrupt the execution of a bundle and start the next bundles in the queue until all bundles are interrupted. Due to this mechanism, threshold synchronization is achieved with a minimum amount of time. It is usually designed for communication between scalar processors via shared memory.

Key words: Computer, Information technology, digital signal processing, mathematical models, network.

Introduction

The multilevel memory hierarchy allows access to global data through the first and second level caches. The second level of the cache is shared between data and texture units, while the first level is shared between the two multiprocessors and is intended for data only. When calculating on a multiprocessor, in addition to data from global memory, two additional types of memory are used: constant memory and shared memory. The constant memory is read-only. It is stored in video memory and cached on the 8 KB multiprocessor (the total size for all multiprocessors is limited to 64 KB; the constant memory is the same for all multiprocessors). The latency when accessing it can be the same time as for accessing global memory in case of a cache miss, but if there is a cache hit, then the access will be performed in 2

multiprocessor clock cycles [5]. Shared memory is intended for reading / writing and is organized in the form of memory banks (16 or 32), each of which can be accessed in 2 clock cycles. The entire bundle gets the request to access the shared memory[11]. In this case, requests may arise immediately to one bank of shared memory, which will entail a conflict and ordering of requests into the queue with an increase in the access time to bank data for the entire bundle. Therefore, it is important to take into account the coherence of requests when creating algorithms [1]. The amount of shared memory is also limited (16 KB), but in later versions of GPUs (Compute Compatibility 2.x) it can be configured depending on the task. In the process of studying various types of access to memory caches, it was revealed that the required number of cycles can vary significantly [10].

The next architectural solution for NVIDIA graphics processors was GT200, presented in June 2008 [9]. The main differences from the G80 in terms of general calculations are:

1) the number of TPCs (Thread Processing Cluster, multiprocessor clusters) has increased;
2) in each TPC, the number of multiprocessors has increased to 3 per cluster, while the L1 cache has also increased;
3) the number of registers for program instructions has doubled;
4) each multiprocessor has a block for computing operations on double precision floating point numbers.

Thus, this architecture made a breakthrough in the number of scalar processors on the graphics adapter and the amount of data processed, but the structural elements and their arrangement remained the same.

Unlike the GT200 architecture, the third generation Fermi architecture [8], presented in September 2009, has significantly redesigned the structural diagram of the GPU. The central element of the architecture is the L2 cache, which is used to access video memory. Accordingly, its volume has also increased significantly. Streaming multiprocessors are formed around this cache, which have also changed from previous architectures:

1) the number of scalar processors has increased to 32, moreover, they are optimized for working with 64-bit data;
2) it became possible to execute two competing bundles of streams on one multiprocessor;
3) the number of blocks for calculating transcendental functions has increased to 4;
4) it became possible to manage the amount of shared memory and the first-level cache for data;
5) there were blocks for calculating data addresses located in a single address space.

The heterogeneous CUDA (Compute Unified Device Architecture) model is used to program NVIDIA GPUs [3, 12]. CUDA includes a superscalar parallel computing programming model, as well as libraries and compiler extensions for several high-level languages. The general scheme of CUDA operation.

The device that is the main one in the computing system (CPU) is called the Host. It runs the main sequential program, which transfers control to the parallel computing device Device (GPU) to implement parallel computations. The program that Device runs is called the Kernel. The kernel is developed in the same language in which the sequential program (C / C ++) is implemented using special language additions. Parallel execution on a Device is implemented due to threads combined into blocks. The blocks, in turn, are combined into a (Grid) section, which must completely cover the data processed by the kernel. In order for the Device to process any data, it is necessary to transfer it to the Device memory, then get the result by copying the data in the opposite direction.

Objective Statement

Obviously, not all computer vision algorithms can be parallelized on GPUs. Any artificial computer vision system, regardless of its area of application, should include the following typical stages of work:

1) image acquisition (photo or video filming);
2) preliminary processing;
3) highlighting characteristic features;

4) detection or segmentation;

5) high-level processing.

Almost all stages can be realized with the help of parallel computer vision algorithms executed on modern parallel computing devices. Some of them can use data parallelism, which is advisable to use on the general-purpose GPUs discussed above. Consider the main groups of computer vision algorithms using data parallelism (this classification is a generalization of groups from):

1) image transformation algorithms - input and output data are two-dimensional images, the coordinates of the output image element differ from the coordinates of the input element. These algorithms include: affine transformations, coordinate system transformations, etc .;

2) filtering algorithms - input and output data are two-dimensional images. Each pixel in the output image is the result of an operation on a group of pixels in the input image that fall into a window of a certain size (filter). Filters can be represented by various mathematical devices: cellular automata, neural networks, functions, etc .;

3) statistical algorithms - input data are two-dimensional images, output data are arrays with statistical information. Each element of the original data can be used for statistical calculations if it meets the requirements of the algorithm. An example of such algorithms can be the calculation of histograms, Hook's transformation, grouping of elements, etc. Parallelization of such algorithms requires communication between processors to combine the accumulated statistical data;

4) recursive algorithms - input and output data are two-dimensional images. Each element of the original image contributes to the formation of all the image elements in the output. These algorithms include: calculation of the integral of the image, distance transformation, etc. This type of algorithms is very difficult to parallelize, since in most cases, computation of a single output item requires the result of previous input items.

One of the basic features of orthogonal bases is presence of fast algorithms for definition of spectral factors. Fast algorithms allow to reduce quantity of arithmetic operations and volume of necessary memory. The increase in speed is as a result reached at use of orthogonal bases for digital processing signals [1, 2, 3, 4].

Methodology

We write down the formula of direct and return fast spectral transformations for sequence of readout of a signal{ xi } for any valid orthogonal piecewise-constant basis

$$C_k = \frac{1}{2^p} \sum_{i=0}^{n-1} x(i) \cdot \varphi(k,i) \qquad (1)$$

$$X_i = \sum_{k=0}^{n-1} C_k \cdot \varphi(k,i) \quad , \qquad (2)$$

where k = number of spectral coefficients,

i = number of an element of sequence of the valid readout.

In this graph, the continuous lines correspond to operations of addition, while the hatch lines are operations of subtraction. Entrance readout is denoted with X0, X1, ... , X15, and results are denoted with C0, C1, C2 ... , C15

The analysis of computational methods of factors in various bases has shown, that fast algorithms for calculation of factors exist only for piecewise-constant and piecewise-linear bases. Algorithms of calculation of factors in piecewise-quadratic bases have not been developed.

We investigate a question how algorithms of fast transformations in bases of orthogonal piecewise-constant functions can be adapted for calculation of factors in piecewise-linear bases. Known formulas Fourier -

https://procedia.online/ ISSN-2795-563X

Haar [1, 5], Fourier - Harmut using integrals of a kind:

$$C_0 = \int_0^1 x(r)\,dr \cong \sum_{i=0}^{n-1} \int_{hpj} x(r)\,dr$$

$$C_k = \int_0^1 x(r)\cdot har_k(r)\;\; dr = \sum_{i=0}^{n-1} har(i) \int_{hpj} x(r)\;\; dr$$

$$i = 1,2,\ldots,\; n,\;\; j = 0,1,\ldots,2^{p-1} \qquad (3)$$

applicable only in the event that transformable signals x (r) belong to metric space L2 [0,1).

The algorithm of calculation of factors does not possess property of fast transformation and, besides if necessary to receive values of factors in локализуемых bases it is possible to use directly operations with final differences.

For example, factors in basis Shauder are calculated on the basis of transformations

$$\Delta f_i = \sum_{k=0}^{n-1} C_k \cdot Shd_k(x_{i+1}) - \sum_{k=0}^{n-1} C_k \cdot Shd_k(x,i) =$$

$$= \sum_{k=0}^{n-1} C_k \left(\int_{hk} har_k(r)dr - \int_{hk} har_k(r)dr \right) = \frac{1}{2^p} \sum_{k=0}^{n-1} C_k\, har_k(x_i) \qquad (4)$$

For the second order of basic functions factors of Harmut's fast transformation C2 and C3 are calculated by grouping the sums of final differences under formulas:

$$C_2 = \left(\sum_{j=0}^{n/4-1} \Delta f_i - \sum_{j=n/4}^{n/2-1} \Delta f_j \right) - \left(\sum_{j=n/2}^{3n/4-1} \Delta f_i - \sum_{j=3n/4}^{n-1} \Delta f_j \right);$$

$$C_3 = \left(\sum_{j=0}^{n/4-1} \Delta f_j - \sum_{j=n/4}^{n/2-1} \Delta f_j \right) - \left(\sum_{j=n/2}^{3n/4-1} \Delta f_j - \sum_{j=3n/4}^{n-1} \Delta f_j \right)$$

Other factors for $P \geq 2$, $k \geq 4$ are calculated as the sum of a difference of a following view:

Conclusions

If the GPU is the Device, then the size of the partition and blocks is limited. Each block is executed on a separate multiprocessor independently of other blocks. Therefore, the width and height of the block are determined by the maximum number of simultaneously processed threads on the multiprocessor.

The numerical experiments allow us to draw a conclusion that the number of zero coefficients at digital processing of signals received as a result of bench tests in Haar and Harmut's piecewise-quadratic bases ranges from 5 % up to 17 %, when processing the geophysical signals received as a result magnetic exploration we get values ranging from 5 % up to 25 %, and while processing elementary functions (and also functions consisting of their combinations) this parameter gives us value from 10 % up to 70 % with an accuracy of 10-4-10-6. It is established, that decomposition (4) allows receiving high speed in Haar's basis and the big factor of compression in Harmut's basis. Also as a result of researches it is revealed, that with increase in quantity of readout function N, the values of factors decreases on exponential law.

List of References

1. Rakhimov BS, Mekhmanov MS, Bekchanov BG. Parallel algorithms for the creation of medical database. J Phys Conf Ser. 2021;1889(2):022090. doi:10.1088/1742-6596/1889/2/022090

2. Rakhimov BS, Rakhimova FB, Sobirova SK. Modeling database management systems in medicine. J Phys Conf Ser. 2021;1889(2):022028. doi:10.1088/1742-6596/1889/2/022028
3. Rakhimov B, Ismoilov O. Management systems for modeling medical database. In: ; 2022:060031. doi:10.1063/5.0089711
4. Rakhimov BS, Khalikova GT, Allaberganov OR, Saidov AB. Overview of graphic processor architectures in data base problems. In: ; 2022:020041. doi:10.1063/5.0092848
5. P. P. Kudryashov Algorithms for detecting a human face for solving applied problems of image analysis and processing: author. dis. Cand. tech. Sciences: 05.13.01. - M, 2007.
6. Tanenbaum E. Modern operating systems. 2nd ed. - SPb .: Peter, 2002 .-- 1040 p .: ill.
7. Forsyth DA, Pons, J. Computer vision. Modern approach / D.A. Forsyth, J. Pons: Trans. from English - M .: Publishing house "Williams", 2004. - 928 p .: ill. - Parallel. tit. English
8. Frolov V. Solution of systems of linear algebraic equations by the preconditioning method on graphic processor devices
9. Brodtkorb A.R., Dyken C., Hagen T.R., Hjelmervik J.M., Storaasli O.O. State-of-the-art in heterogeneous computing / A.R. Brodtkorb, C. Dyken, T.R. Hagen, J.M. Hjelmervik, O.O. Storaasli // Scientific Programming, T. 18, 2010. - S. 1-33.
10. Zaynidinov H., Mallayev O., Kuchkarov M. Parallel algorithm for modeling temperature fields using the splines method 2021 IEEE International IOT, Electronics and Mechatronics Conference, IEMTRONICS 2021 - Proceedings, 2021, 9422645
11. Zaynidinov H., Makhmudjanov S., Rajabov F., Singh D. IoT-Enabled Mobile Device for Electrogastrography Signal Processing Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2021, 12616 LNCS, стр. 346–356
12. Zaynidinov H.N., Yusupov I., Juraev J.U., Singh D. Digital Processing of Blood Image by Applying Two-Dimensional Haar Wavelets Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in bioinformatics), 2021, 12615 LNCS, стр. 83–94