Parallel algorithms for the creation of medical database systems

Bakhtiyar S Rakhimov¹, Mukhiddin S Mekhmanov and Bakhtiyar G Bekchanov
Urgench branch of Tashkent Medical Academy, Urgench, Uzbekistan

¹E-mail: bahtiyar1975@mail.ru

Abstract. Analysis of parallel algorithms for graphics processors allows you to determine the bottlenecks of the algorithm that affect its performance on a particular computing system. Algorithms can be analyzed both at the AGM level and at the level of the CPU-GPU system. Despite the fact that the model does not reflect the overlap of the execution of computational operations and operations of access to the global memory, it estimates the upper limit of the execution time of the algorithm on the AGM, in any parallel algorithms for the AGM it is necessary to reduce this parameter of the algorithm due to the large number of clock cycles spent by the multiprocessor to implement global memory access operations.

1. Introduction
When analyzing a parallel algorithm at the level of a graphics multiprocessor, you can use the methods of analyzing parallel algorithms for the PRAM model, which are based on the working and step complexity of the algorithm. A graphics multiprocessor contains a finite number of scalar processors, and even though software can emulate significantly more processors, you need to pay special attention to the operational complexity of the algorithm, which should be sufficient to cover the cost of accessing the global memory of the graphics processor. That is why, at this level, it is important to evaluate the algorithm's access to the global memory of the graphics processor.

2. Parallel algorithms for the creation of medical databases
Thus, according to the proposed AGM model, when analyzing each stage of the algorithm, it is necessary to estimate the following set of step parameters:

- the working complexity of the algorithm for the AGM. With small amounts of processed data;
- the complexity of access to the global memory of the graphics processor of the parallel algorithm for the AGM. The estimation of this parameter is important for any algorithm on the AGM. When evaluating the execution time of the algorithm, the number of accesses to the global memory is estimated for all AGM processors separately from the operational complexity. When comparing the complexity of access to the global memory of the graphics processor and the operational complexity of the algorithm, information is required on the parameters of the latency of access to the global memory and the speed of the scalar processor, which significantly affect the decision to choose one or another PRAM algorithm at a particular stage of the parallel algorithm for the graphics processor;
• the amount of data transferred at each stage of the algorithm between the computer's RAM and
the global memory of the graphics processor, etc. These parameters are also involved in deciding
on the choice of one or another PRAM algorithm at the th stage, since with low RAM-GRAM
bandwidth and high speed of multiprocessors with low operational complexity, the transfer of
computations to the GPU may not be beneficial for performance.

When analyzing parallel algorithms for graphics processors at the level of the CPU-GPU computing
system, it is necessary to operate with the total parameters of the parallel algorithm at all its stages:

• total working complexity:

\[ W(N) = \sum_{i=1}^{B(N)} W^i(N) \]  

(1)

• total complexity of accessing global GPU memory:

\[ R(N) = \sum_{i=1}^{B(N)} R^i(N) \]  

(2)

• the total amount of data transferred between the computer's RAM and the global memory of the
graphics processor \( N_{HD}(N) \) and \( N_{DH}(N) \), found by formulas (1) and (2).

With the help of the proposed total parameters of parallel algorithms, it is possible to make a decision
on the choice of a specific parallel algorithm for a specific graphics processor, or to leave the calculations
entirely on the central processor.

The total parameters of parallel algorithms for graphics processors are used to make decisions in
particular cases of optimization of algorithms for a specific graphics processor. In general, using them
is rather difficult to choose the most efficient algorithm. Decision making in real time should be based
on time estimates. Therefore, the comparison of parallel algorithms for graphics processors can be
carried out according to the following indicators: evaluation of the execution time, acceleration and
efficiency of computations.

Before comparing parallel algorithms for graphics processors, it is necessary to make a decision on
the choice of specific PRAM algorithms at each stage. This decision should be based on an estimate of
the execution time of the PRAM algorithm for AGM, taking into account the transfer of data between
RAM and GRAM according to the formula:

\[ T_{GPU_q}^i(N) = \frac{N_{HDq}^i(N)}{S_{HD}^i} + T_{G_q}^i(N) + \frac{N_{DHq}^i(N)}{S_{DH}^i}, \]  

(3)

where \( q \) - is the number of the PRAM algorithm at the th stage of the parallel algorithm for the
GPU;

\[ T_{G_q}^i(N) \] - estimation of the execution time of the \( q \) - th PRAM algorithm of the \( i \) - th stage
according to formula (2);

\[ N_{HDq}^i(N), N_{DHq}^i(N) \] - the amount of data transferred between RAM and GRAM for the \( q \) -
th PRAM algorithm of the \( i \) - th stage.

If \( Q_i \) stands for the total number of developed PRAM algorithms for the \( i \) - th stage, then, comparing
these algorithms by estimating the execution time according to formula (3), we can decide
on the choice of the optimal one for a particular graphics processor:

\[ T_{GPU}^i(N) = \min_{q=Q_i} T_{GPU_q}^i(N). \]  

(4)
The estimation of the execution time of the parallel algorithm on the GPU should be carried out according to the formula (1) with the already selected PRAM algorithms for each stage. Using formula (4) and (2), we can find the total execution time of the algorithm \( j \):

\[
T_{GPU}^{j}(N) = \sum_{i=1}^{B_i(N)} T_{GPU}^{i}(N).
\]  

(5)

The criterion for choosing the best parallel algorithm from the \( J \) developed algorithms for the GPU will be the minimum execution time estimate:

\[
T_{GPU}(N) = \min_{j \in J} T_{GPU}^{j}(N).
\]  

(6)

Where \( T_{GPU}^{j}(N) \) - is an estimate of the running time of the \( j \) - th parallel algorithm for the graphics processor.

Parallel algorithms for GPUs can be compared by the amount of acceleration of parallel computations on the GPU compared to the serial version of the algorithm. To estimate the relative acceleration, you can use the following formula:

\[
S^{j}(N) = \frac{T_{S}(N)}{T_{GPU}^{j}(N)}.
\]  

(7)

Then the criterion for choosing the best parallel algorithm from the developed algorithms for the GPU will be the maximum acceleration estimate:

\[
S(N) = \max_{j \in J} S^{j}(N).
\]  

(8)

Since for graphics processors multiprocessors are the main computational units, to compare algorithms, you can use an estimate of the efficiency of using multiprocessors:

\[
E^{j}(N) = \frac{T_{S}(N)}{P \cdot T_{GPU}^{j}(N)} = \frac{S^{j}(N)}{P}.
\]  

(9)

Then the criterion for choosing the best parallel algorithm from \( J \) of the developed algorithms for the graphics processor will be the maximum approximation of the estimate of the efficiency of using multiprocessors to unity:

\[
E(N) = \min_{j \in J} (|E^{j}(N) - 1|).
\]  

(10)

3. Conclusions

In summary, the demand for parallel algorithm is growing day by day to address large-scale parallel algorithm or to facilitate user access to data management. Because parallel algorithm are not only used for data entry and storage, they also describe their structure: file collection supports logical consistency; provides data processing language; restores data after various interruptions; parallel algorithm systems allow multiple users to work in parallel.

The main requirement for parallel algorithm systems is to safely store external data and to respond to the request of the user to satisfy it. This requires the completeness of the information stored in the medical database to maintain the integrity of the data.
References

[1] Carl de Boor 2001 *A practical Guide to splines. Department of computer science University of Wisconsin* (USA: Madison)

[2] Wang Ren-Hong 2001 *Multivariate spline functions and their applications* (Netherlands: Kuliver Academic Press)

[3] Rakhimov B S, Allabeganov O R and Saidov A B 2020 Processor means for the spectral analysis of medical signals on the of polynomial walsh bases epra *International Journal of Research and Development (IJRD)* 5(7) pp 10-1

[4] Rakhimov B S, Ismoilov O I and Ozodov R O 2017 Russian “Software and automation of forensic examination” *Methods of science Scientific and practical journal* 11 pp 28-30

[5] Rakhimov B S 2017 Russian “Information technologies in medical education” *Methods of science Scientific and practical journal* 12 pp 25-7

[6] Pen U 2000 *Application of Wavelets to Filtering of Noisy Data* (Oxford: Oxford University Press)

[7] Balakumaran Ashok V, Gowrishankar T, Vennila C, Kumar A and Nirmal A The Fast Haar Wavelet Transform for Signal & Image Processing. *International Journal of Computer*

[8] Zaynidinov H, Zaynudinova M and Nazirova E 2018 Digital processing of two-dimensional signals *International Journal of Advanced Trends in Computer Science and Engineering* 9(4) 5259-65 5265

[9] Zayniddinov H N and Mallayev O U 2019 Paralleling of calculations and vectorization of processes in digital treatment of seismic signals by cubic spline *IOP Conference Series: Materials Science and Engineering* 537 https://doi.org/10.1088/1757-899X/537/3/032002

[10] Zaynidinov H N, Yusupov I, Juraev J U and Jabraev J S 2020 Applying two-dimensional piecewise-polynomial basis for medical image processing *International Journal of Advanced Trends in Computer Science and Engineering* 9(4) 5259-65 156

[11] Zaynidinov H N, Mallayev O U and Yusupov I 2020 Cubic basic splines and parallel algorithms *International Journal of Advanced Trends in Computer Science and Engineering* 9(3) pp 3957-60

[12] Zaynidinov H, Juraev J and Juraev U 2020 Digital image processing with two-dimensional Haar wavelets *International Journal of Advanced Trends in Computer Science and Engineering* 9(3) pp 2729-34