Estimation of the characteristics of complex objects using big data technologies

Anatoly D. Khomonenko[a,b], Igor A. Molodkin[a] and Alena I. Zimovets[b]

[a] *Emperor Alexander I St. Petersburg State Transport University, 9 Moskovsky pr., St. Petersburg, 190031, Russia*
[b] *Military Space Academy named after A.F. Mozhaisky, st. Zhdanovskaya, 13, St. Petersburg, 197198, Russia*

Abstract
A brief description of the main modern approaches and tools used for processing big data (cloud storages, NoSQL databases, Hadoop MapReduce and Disco platforms, artificial intelligence and deep learning, parallel programming and processing) is given. Calculated estimates of the increase in the performance of calculations are obtained using the example of predicting the motion parameters of small spacecraft using the CUDA software and hardware architecture of parallel computing using graphics processors from Nvidia. The acceleration of the time for calculating using GPU in comparison with CPU is 27.5%.

Keywords 1
Big data, complex objects, parallel computing, CUDA.

1. Introduction

At the present stage of the development of science and technology in the era of globalization, the main trend is a significant increase in the volume of information and the related problems of their storage and processing. The active implementation of Big Data technologies contributes to the growth of requirements for productivity, accuracy and optimization of methods and monitoring tools.

Advanced computing systems are designed to solve multichannel information processing in real time, functional and technical control of data processing equipment.

Using computer systems, the following tasks are solved [1]:

- automatic processing of information from receiving means, which is divided, as a rule, into primary (processing of signal information) and secondary (processing of trajectory information);
- automated control over the operation of the hardware complex (including position control, power distribution, etc.);
- display and documentation of processing results;
- ensuring interaction with other systems for collecting and processing information;
- control, diagnostics, optimization of work and ensuring the stability of the functioning of funds.

Nowadays, Big Data has ceased to be something new and unknown, but its importance has not only not diminished, but continues to grow every year. Currently, there are a number of approaches to implementing big data processing [2]:

1. Cloud storage, in which data storage and processing become faster and more economical (compared to the cost of maintaining a data center and staff).
2. NoSQL databases - allow you to work with unstructured and semi-structured data, as well as with the wide capabilities of

Models and Methods for Researching Information Systems in Transport, Dec. 11-12, St. Petersburg, Russia
EMAIL: khomon@mail.ru (A.D. Khomonenko); molodkin@pgups.ru (I.A. Molodkin); zim_alenka@rambler.ru (A.I. Zimovets)
ORCID: 0000-0003-0411-258X (A.D. Khomonenko);

distributed systems and horizontal scaling. In modern conditions of development of the information space, this technology has advantages in terms of flexibility and efficiency of data access.

3. Tools for processing big data:

Hadoop MapReduce is a distributed computing model used for parallel computing on very large, up to several petabytes, datasets in computer clusters.

Disco – Also performs the task of distributed batch processing of large amounts of data.

4. Artificial Intelligence and Deep Learning – a machine learning technology that mimics the structure and work of the human brain, is the best suited for processing large volumes of constantly changing information.

One of the ways to increase the efficiency of Big Data processing is the use of parallel computing. In information technology, the term "parallel" or "parallelism" is associated with solving complex problems and means that parts of a complex task are performed simultaneously and independently of each other. The goal of any parallelism is to improve the efficiency of the computer at various levels, namely at the levels: tasks, data, algorithms, instructions, bits.

One modern way to implement parallel computing is the use graphics processing units (GPU), which are superior to central processing units (CPU) at the expense of graphics accelerators. The advanced technology for implementing this approach is the CUDA platform from Nvidia. The peculiarity of computing on GPU is that, unlike CPU, they uniquely designed to handle multiple threads [3].

Any graphics processor is a multiprocessor consisting of a certain number of computing clusters with many arithmetic-logic devices in each.

Another important feature is the organization of memory in CUDA and the distribution of the threads access level to various parts of it.

Thus, CUDA platform allows the programmer to use all possible resources of the GPU and direct them to the implementation of high-performance application.

Nvidia CUDA technology is well suited for solving a wide range of tasks and significantly facilitates software development due to a developed set of extensions for the C and C++ language [3, 4] that allow you to express both data parallelism and task parallelism at the level of small and large structural units. However, it should remembered that GPU couldn't replace CPU due to the peculiarities of their architecture and operation.

2. Recognition and prediction of characteristics of complex objects using parallel computing

2.1. Recognition of characteristics of CO

Let us consider the efficiency of parallelization by the example of a complex object recognition algorithm.

Given:

- fuzzy inference system containing seven linguistic variables with three or five terms, 16875 rules of fuzzy productions and one output variable.
- investigated data.

The structural and logical scheme of recognition in this case is shown in Figure 1.

In the diagram shown, the rule base Ψ and the data for analysis $K = \{k_j,\ j = 1,\ldots, K\}$ are given as the initial data. The rule base based on the results of the analysis of photometric information of the multichannel monitoring telescope (MMT) [5] and includes the following characteristics:

L is the average reduced gloss CO;
ESS – effective scattering area;
H_p – perigee height;
H_a is the height of the apogee;
i is the inclination of the orbit of the SO;
τL is the brightness periodicity of the CO;
τ – Period of circulation of the CO.

It proposed to implement parallelization at the stage of checking each complex object with rules.

Let the number of processors be equal to p and n – the number of operations to parallel processing. In this case, the parallel computation time will be [6]:

$$T_{parall} = \tau_A * \frac{n}{p} + \tau_A * \log_2 p.$$

In this formula, the number of terms in the progression defined by the expression $\log_2 p + 1$.

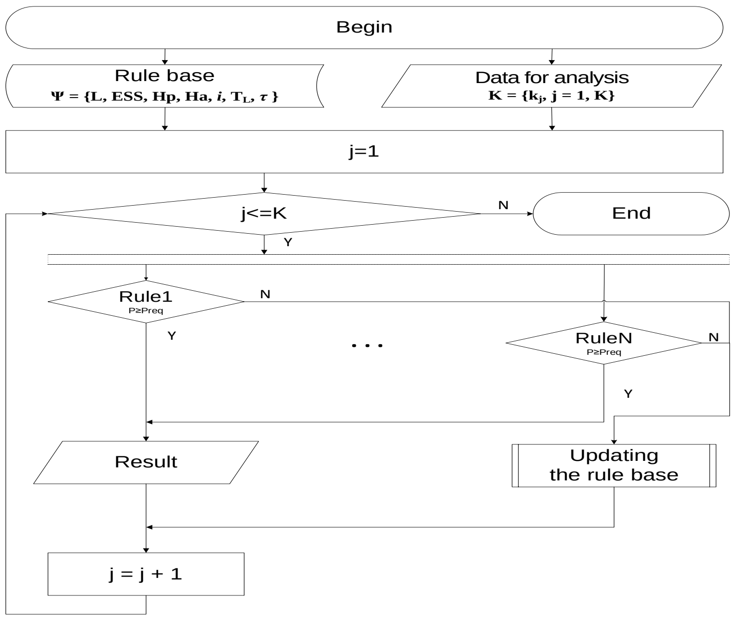

Figure 1: Structural and logical diagram of the algorithm for recognizing complex objects

Thus, efficiency and acceleration indicators are calculated:

$$S_p = \frac{n\tau_A}{\tau_A \frac{n}{p} + \tau_A \log_2 p} = p\,\frac{1}{1 + \frac{p}{n}\log_2 p};$$

$$E_p = \frac{1}{1 + \frac{p}{n}\log_2 p}.$$

Thus, for p = n / log₂n and n/p is an integer, we have:

$$S_p = \frac{p}{2E_p} = 50\%.$$

In this case, the average degree of parallelism is

$$\frac{1}{\log_2 p + 1} * p * \frac{1 - (0{,}5)^{(\log_2 p + 1)}}{1 - 0{,}5}.$$

In particular, for $p = 2$, the average degree of parallelism is 1.5.

Figure 2 shows that the optimal time result with an increase in the number of rules is given by using two processors.

Table 1 presents the results of experiments showing the advantage of parallel over sequential computation when the number of rules increases.

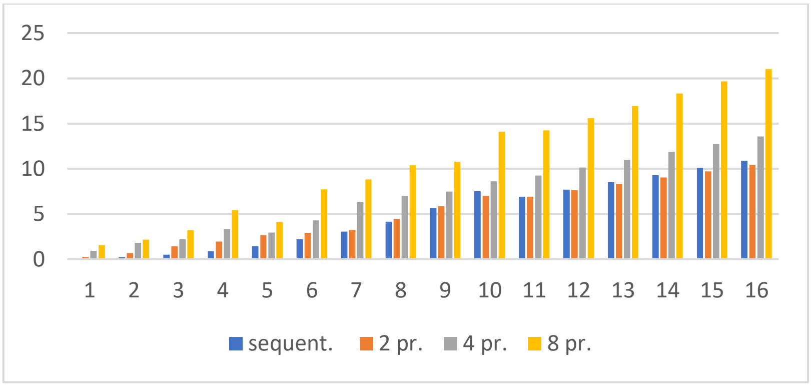

Figure 2: Time dependence, calculation on the number of processors

Table 1
Results of experiments

Number of rules	**Sequential algorithm**	**Parallel algorithm**					
		2 pr.		4 pr.		8 pr.	
		Time	Acceleration	Time	Acceleration	Time	Acceleration
1000	0,0435	0,2476	0,1757	0,9320	0,0467	1,5735	0,0277
2000	0,2079	0,6837	0,3041	1,7999	0,1155	2,1591	0,0963
3000	0,4849	1,4034	0,3455	2,2136	0,2191	3,1953	0,1518
4000	0,8729	1,9455	0,6220	3,3237	0,2626	5,4309	0,1607
5000	1,4324	2,6647	0,7363	2,9331	0,4884	4,1189	0,3478
6000	2,1889	2,8999	0,8214	4,2911	0,5101	7,7373	0,2829
7000	3,0424	3,2364	1,0491	6,3273	0,4808	8,8255	0,3447
8000	4,1497	4,4621	1,2822	6,9931	0,5934	10,3898	0,3994
9000	5,6218	5,8340	1,2599	7,4747	0,7521	10,7636	0,5223
10000	7,5116	6,9902	1,2875	8,5968	0,8738	14,0951	0,5329

The results given in Figure 2 and Table 1 show that when organizing parallel computations, it is advisable to determine the optimal number of parallel computational threads in order to avoid unnecessary overhead.

2.2. Prediction of characteristics of CO

Estimates of the increase in the performance of calculations on the example of predicting the motion parameters of small spacecraft [7] using the CUDA software and hardware architecture.

The characteristics of each object are a two-line set of TLE parameters [8] (Figure 3), which includes the following parameters: year and time of the epoch, orbit number at the time of the epoch, acceleration, deceleration coefficient, Keplerian orbital elements, and checksums for authentication data.

METEOR 1-29
1 11251U 79005A 19081.29151034 .000010
2 11251 97.6714 339.2672 0014582 306.827

Figure 3. Example of a TLE data format

For verification, we used the formula for the lateral movement of an object [7]:

$$\zeta = \zeta_0 \cos(\omega t) + \zeta'_0 \omega^{-1} \sin(\omega t) + \omega^{-1} \int_0^t W(\tau) \sin[\omega (t-\tau)] d\tau, \quad ,$$

where ζ_0, ζ_0’are the initial parameters of the object's motion, τ is the time satisfying the condition $0 \le \tau \le t$, ω is the angular velocity of motion, W (t) is the control acceleration. The lateral motion was calculated for the angular position of the orbit from 0 to 359, the altitude of the circular orbit from 100 to 500 km (for a greater computational load) and 3 different variants of the control action. Thus, 360 * 400 * 3 = 432,000 double values were calculated.

The types of control actions W (t) are shown in Figure 4.

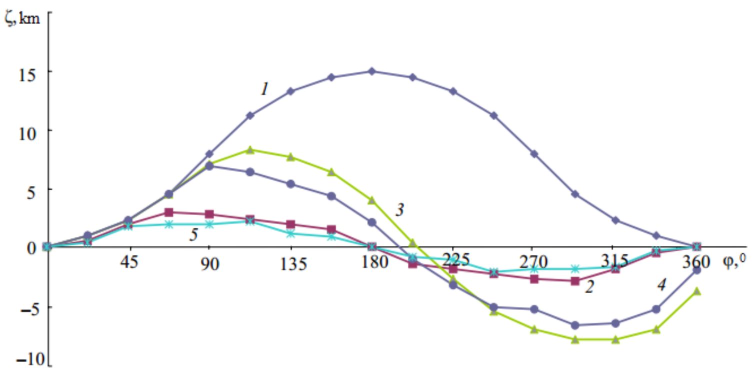

Figure 4: The types of control actions W (t)

A fragment of the calculation of the software package is shown in Listing 1. The results obtained make it possible to draw more prompt conclusions about the influence of the control action on the space object.

Table 2 presents the Diagnostic calculation data with using CPU and GPU .

Listing 1:

```
#define BLOCK_SIZE 32
#define PI 3.14159265
//#define OMEGA 0.09294116717 //422 km
//#define R = 422
#define ZETA0 0
#define ZETA0P 1
#define MINH 100
#define MAXH 500
using namespace std;
void cpu_calc_4(double* zeta) {
  for (int deg = 0; deg < 360; deg++) {
    for (int r = MINH, int i = 0; r <= MAXH; r++, i++) {
      double OMEGA = (sqrt(3.98602 * 100000)) / (6371 + r);
      int idx = i * 360 + deg;
      if (deg >= 0 && deg <= 60) zeta[idx] = ZETA0 * cos(deg * PI / 180.0) + ZETA0P * (1.0 /
OMEGA) * sin(deg * PI / 180.0) + (1.0 / OMEGA) * (cos(OMEGA * PI / 180.0) / 100.0 * OMEGA);
      else zeta[idx] = ZETA0 * cos(deg * PI / 180.0) + ZETA0P * (1.0 / OMEGA) * sin(deg * PI /
180.0) + (1.0 / OMEGA);}}}
__global__ void gpu_calc_4(double* zeta) {
  double deg = blockIdx.x * blockDim.x + threadIdx.x;
  for (int r = MINH, int i = 0; r <= MAXH; r++, i++) {
    double OMEGA = (sqrt(3.98602 * 100000)) / (6371 + r);
    int idx = i * 360 + deg;
    if (deg >= 0 && deg <= 60) zeta[idx] = ZETA0 * cos(deg * PI / 180.0) + ZETA0P * (1.0 /
OMEGA) * sin(deg * PI / 180.0) + (1.0 / OMEGA) * (cos(OMEGA * PI / 180.0) / 100.0 * OMEGA);
    else zeta[idx] = ZETA0 * cos(deg * PI / 180.0) + ZETA0P * (1.0 / OMEGA) * sin(deg * PI /
180.0) + (1.0 / OMEGA);}}
```

Table 2
Diagnostic calculation data

Type	Time (%)	Time	Calls	Avg	Min	Max	Name
GPU activity.	33.65	1.9699ms	1	1.9699ms	1.9699ms	1.9699ms	gpu_calc_5 (double")

	25.66	1.4667ms	1	1.4667ms	1.4667ms	1.4667ms	gpu_calc_4(double")
	25.62	1.4642ms	1	1.4642ms	1.4642ms	1.4642ms	gpu_calc_6(double")
	16.27	952.40US	3	317.47US	264.40US	370.16us	CUDA memcpy HtoH

With the help of CPU, the calculation of the motion parameters of a small spacecraft required 0,004 sec. With the help of GPU, the calculation of the motion parameters required 0,0029 sec. Thus, the acceleration of the time for calculating the parameters of the orbit of a small spacecraft using GPU in comparison with CPU is 27.5%.

3. Conclusion

For an in-depth study of the state of research on the topics touched upon by us, it is recommended to familiarize yourself with works [8-12]. The approaches to big data processing considered in this article in the recognition and prediction of the characteristics of complex objects based on the use of parallelization with the help of GPUs show a very noticeable acceleration of computations. In particular, by the example of solving the problem of determining the parameters of the motion of small spacecraft using graphic processors, an acceleration of calculations of 27.5% was obtained.

References

[1] Isaev E.A., Kornilov V.V. The Problem of Processing and Storage of Large Amounts of Scientific Data and Approaches to Its Solution. Mathematical Biology and Bioinformatics, 2013 Vol. 8, №1, Pp.49-65

[2] Maikibaeva E.K. “Big Data and Data Mining tasks”, Materials of the I International Scientific and Practical Conference, 2018, p.129-131.

[3] Maltsev N.S., Molodkin I.A., Gilvanov R.G. “Comparative Analysis of Computing Speed on central and Graphic Processor”, Intellectual Technologies on Transport, 2020, #2, p.51-57.

[4] Demidov D., Ahnert K., Rupp K., Gottschling P. “Programming CUDA and OPENCL: a Case Study Using Modern C++ Libraries”, SIAM Journal on Scintific Computing, #35(5), 2012.

[5] Results of the analysis of photometric information on space objects in near-Earth orbits for September-December 2020. Information and Analytical report. http://mmt9.ru/wp-ontent/uploads/report/MMT_2020_11.pdf .

[6] Volosova A.V. Parallel Methods and Algorithms, textbook, M.: MADI, 2020, p.176.

[7] Averianov A.V. “Analytical Method for Calculation of Small Space Vehicle Motion Linked to a Basic Spacecraft”, Journal of Instrument Engineering, Vol.52, #4, 2009, p.75-77.

[8] Khomonenko A.D., Zimovets A.I. “The rationale for choosing a data storage model for a space monitoring system”, Automation on Transport, Vol.5, #2, 2019, p.221-232.

[9] Zakharov, A.I., Lokhvitskii, V.A., Starobinets, D.Yu., Khomonenko, A.D. Evaluation of the impact of parallel image processing on the operational efficiency of the Earth remote sensing spacecraft control complex. Sovremennye Problemy Distantsionnogo Zondirovaniya Zemli iz Kosmosa, 2019, 16(1), p. 61–71.

[10] Guzik V.F., Zolotovsky V.E., Muntyan O.A., “Parallelization in Modeling Systems”, Izvestiya TRTU, #3(32), 2003, p.70-74.

[11] Khomonenko A.D., Plyaskin S.P., Zimovets A.I. “About complex objects defining via integration of data from various sources”, MMISR 2019 Models and Methods of Information Systems Research Workshop in the Frame of the Betancourt International Engineering Forum, 2020, p. 46-51.

[12] Klemenkov P.A., Kuznetsov S.D. “Big Data: Modern Approaches to Storage and Processing”, Proceedings of the Institute for System Programming of the RAS (Proceedings of ISP RAS), Vol.23, 2012, p.143-158.