American Research Journal of Mathematics **Original Article**
ISSN 2378-704X **Volume 1, Issue1, Feb-2015**

Solution of Problems of Global Optimization of Functions of Several Variables on the Graphics Processors

Mustafina S[1], Grigoryev I[2]

[1]Professor of Bashkir State University, Sterlitamak, Physics and Mathematics Department, Russia
[2]Assistant Bashkir State University, Sterlitamak, Physics and Mathematics Department, Russia

Abstract: Based on the method of particle swarm algorithm is developed in parallel global extremum search. C language for parallel programming in the system implements a method of particle swarm to the global minimization of functions. The algorithm been tested on a spherical function and Rastrigin function.

Keywords: Global extremum, the method of particle swarm, parallel computing, multiprocessor systems, Nvidia CUDA.

I. INTRODUCTION

Copying the actions of nature, man creates more and more sophisticated optimization algorithms. To create them often are examples from nature, such as the genetic code, or bird behavior modeling migration of fish, cooling of the metal, etc. Currently in production and business optimization algorithms are widely used because they provide an opportunity to save not only money, but also time, which is not always enough.

Consider the problem of global unconditional optimization objective function $f(x)$ in n-dimensional arithmetic space R^n:

$$\min_{x \in R^n} f(x) = f(x^*).$$

To solve this problem was chosen as the method of particle swarm optimization. Particle swarm method was originally developed for the graphic simulation of choreography flock of birds in the future; it has been developed for various areas of applications.

Flocks of birds and schools of fish exhibit fascinating coordinated collective behavior. Birds, moving randomly, looking for their own food while watching someone else and those closer to the food. The ability to gather in flocks has several advantages for individuals: increased efficiency of foraging, predator avoidance is improved, empowering mating. With each individual, using only local information, the whole flock shows variability coherent (coherent) movement, which looks perfectly synchronized.

Is like a flock of motion can be modeled based on three main principles of behavior, in which only the information from its neighbors:

1) to avoid collisions with nearby individuals from flocks;

2) to select the speed according to the speed of individuals moving close;

3) Try to stay at a small distance from the closest individuals.

Particle swarm method belongs to the class of evolutionary behavioral methods for global optimization. Behavioral methods are based on modeling the collective behavior of self-organizing living or non-living systems composed of simple agents. The key ideas of behavioral methods are decentralized interaction of agents, ease of agent behavior.This method, like all algorithms belonging to the family of evolutionary algorithms is stochastic gradient calculation do not require that it can be used in cases where a gradient computation is impossible, or it has a high computational complexity.There are several variations of the method. For example, in the canonical particle swarm method proposed in 1995, in the work of Kennedy, Eberhart [5], at each iteration in determining the position of a

[1] *Corresponding Author : mustafina_sa@mail.ru*

particle is taken into account the following information on the best particle among the "neighbors" of the particles, as well as details of the particle on the iteration when this particle corresponds to the best value of the objective function.

In the method of particle swarm optimization agents are particles in the parameter space of the optimization problem. Each particle has a defined location and speed of the search space and thus characterizes a specific solution. Like birds, moving in the environment in search of food or evading predators, particles pass through the search space, finding high quality solutions.

Each time the particles have a space in the position and velocity vector, which changes at each iteration according to the following formula:

$$\vec{v} \leftarrow \omega \cdot \vec{v} + c_1 \cdot rnd() \cdot (\vec{p}_{best} - \vec{x}) + c_2 \cdot rnd() \cdot (\vec{g}_{best} - \vec{x}),$$

where the coefficient ω, named Juha Shi (Yuhui Shi) and Russell Eberhard inertia factor [3], determines the balance between the breadth of research and attention to sub-optimal solutions found. In case when $\omega > 1$ the particle velocity increases, they fly away and examined more closely space. Otherwise, the particle velocity decreases over time and the rate of convergence in this case depends on the choice of parameters c_1, c_2 - constant acceleration p_{best} - the best point found by the particle g_{best} - the best point of the system passed all the particles $\vec{x}$ - the current position of the particle, and the function $rnd()$ returns a random number between 0 to 1 inclusive.

After calculating the direction vector $\vec{v}$, the particle moves to a point $\vec{x} \leftarrow \vec{x} + \vec{v}$. Based on this best achieved extremum particles and information about the optimal particles in the swarm, if necessary, updated values of the best points for each particle for all particles in the whole [2]. After this cycle is repeated.

In this paper was selected following the termination condition of the algorithm: the work is completed to achieve a certain number of iterations, during which the decision was not improved.

II. DISCUSSION

In the proposed algorithm, it is assumed for computation, the central processing unit (CPU) and graphics processing unit (GPU), i.e., the so-called heterogeneous computing environment [1]. Computations that can be performed independently are performed on the GPU. Areas that cannot be parallelized will be operated by CPU. Run parallel sections of code on the graphics device allow technology CUDA[10].

On the basis of the algorithm implemented the program, approved on spherical function, and Rastrigin function. Figures 1 and 2 shows the results of the particle swarm in the method depending on the number of particles in the swarm (8, 16, 32, 64 and 128 particles) in two-dimensional space.

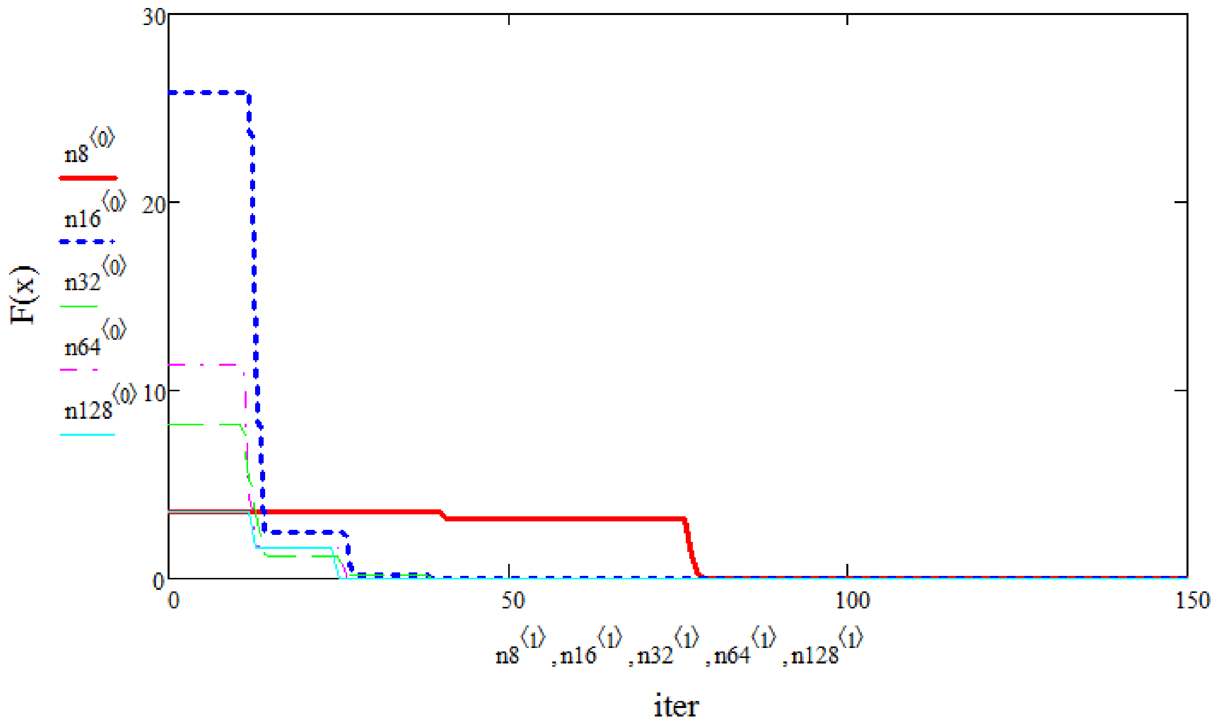

Fig1. *The rate of convergence of the method according to the amount of particles in the swarm for spherical function (n = 2)*

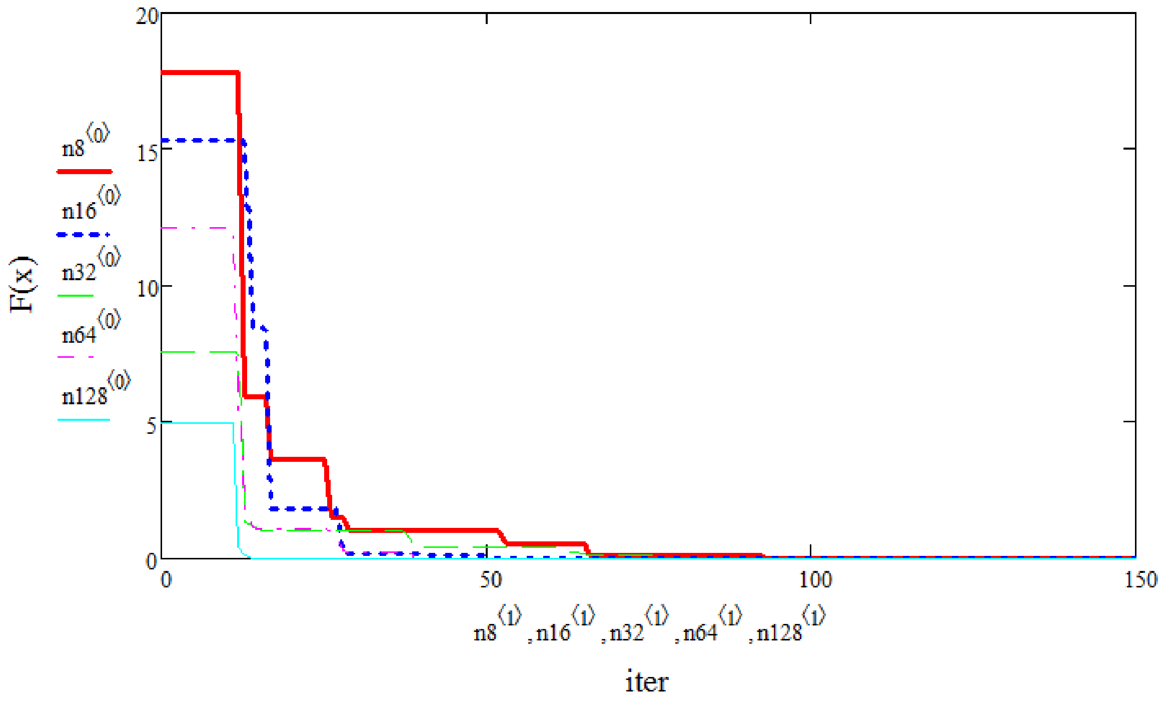

Fig2. *The speed of convergence of the method according to the amount of particles in the swarm for function Rastrigin (n = 2)*

Because this algorithm is largely random search algorithm, the increase in the size of the swarm and the duration of the algorithm (number of iterations) obviously increases the probability of finding the correct solution.

Figures 3 and 4 also show the results of the algorithm depending on the number of particles in the swarm, but now for a three-dimensional space.

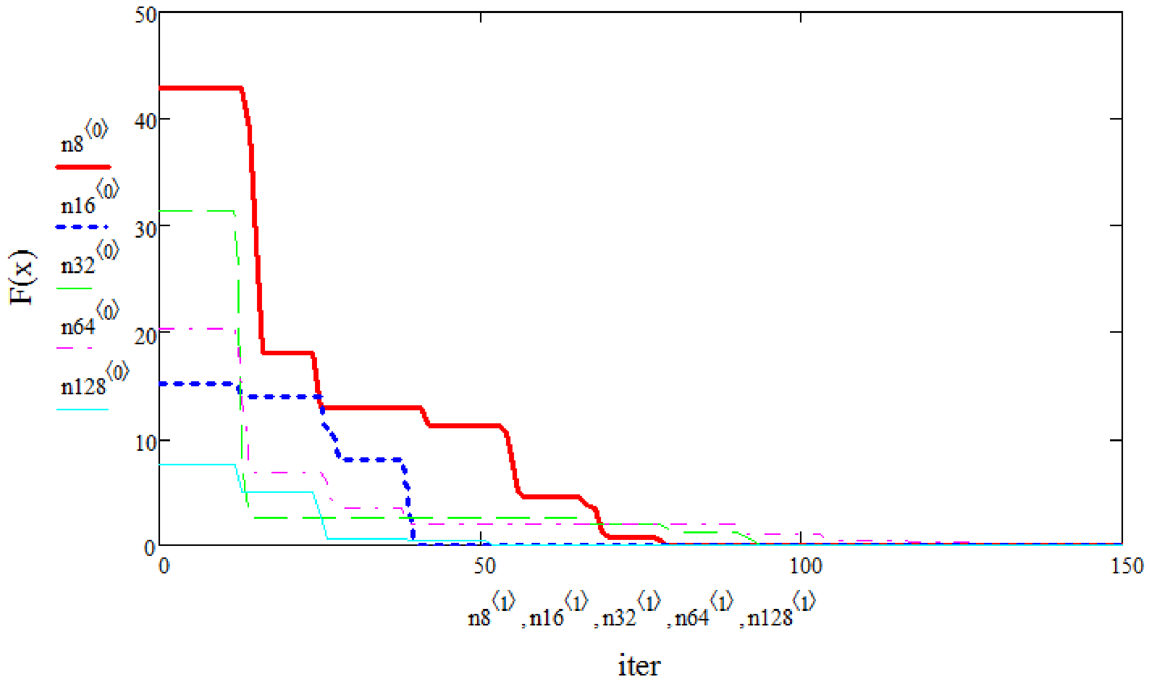

Figure3. *The speed of convergence of the method according to the amount of particles in the swarm for spherical function (n = 3)*

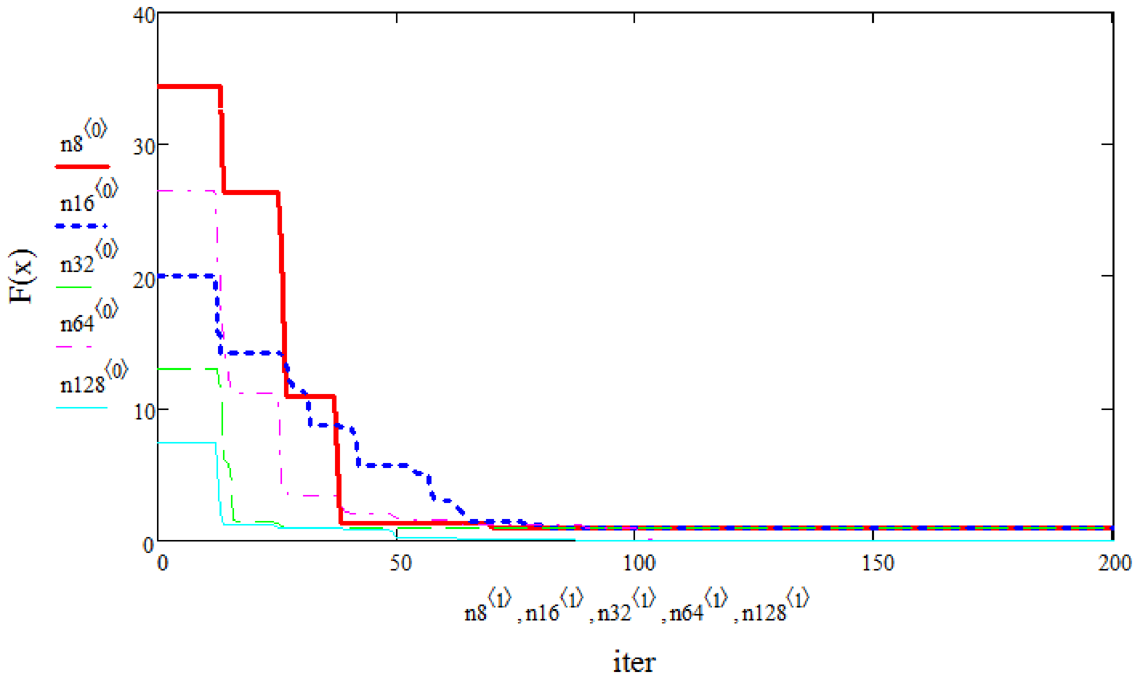

Figure4. *The rate of convergence of the method according to the amount of particles in the swarm for function Rastrigin (n = 3)*

From Figure 2, Figure 4 shows how the accuracy of the solution increases with an increase in the swarm, but at the same time increases significantly during operation. In this regard, it was decided to to implement parallel algorithm for nVidia Cuda.

Most of the known methods of particle swarm are sequential. Parallel methods are little known, and they all appeared after 2004. In this paper presents a method of using the island model of concurrency. The basic idea of this method is as follows: the whole swarm of particles *N* is divisible by *m* Islands (number of computing devices in the system) and the particles belonging to each of the islands, are processed on your graphics processor. After each iteration *k* independent islands are sharing best particles.

To compare sequential and parallel implementations of the algorithm of particle swarm, a series of experiments. Testing was conducted on algorithms computing system with a graphical computing device GeForce GT 520M, CPU Intel Core i5-2410M 2.3 GHz, the operating system Microsoft Windows 7 driver installed NVIDIA CUDA Version 5.5.

Efficiency of parallel computing acceleration is estimated using:

$$S = \frac{T_{CPU}}{T_{GPU}},$$

Where T_{CPU} - while solving the problem on a single processor computer T_{GPU} - a similar time in the solution using a GPU. We emphasize that under acceleration in S this case refers to work faster using GPU performance with respect to calculations performed on a single processor computer. Each program starts at 5 times, was taken as a result of the arithmetic mean of the data.

Table 1 shows the time of the algorithm and the resulting acceleration depending on the size of the swarm for spherical function.

Table1. *Comparative analysis of parallel and serial algorithms of particle swarm for spherical function*

n	N	T_{CPU}	T_{GPU}	Acceleration
2	64	2,906	3,029	0,959
	128	4,573	4,443	1,029
	256	8,907	6,166	1,445
3	64	3,736	3,481	1,073
	256	14,255	4,979	2,68
	1024	57,622	20,96	2,749
	4096	233,917	50.519	4,630
4	64	4,410	3,979	1,108
	256	16,988	7,927	2,143
	1024	67,828	27,861	2,435
	4096	272,351	70,615	3,857

Table2. *Comparative analysis of parallel and serial algorithms of particle swarm for function Rastrigin*

n	N	T_{CPU}	T_{GPU}	Acceleration
2	64	5,44	4,066	1,338
	128	10,615	5,666	1,873
	256	20,964	6,939	3,021
3	64	8,221	2,897	2,838
	256	32,17	5,443	5,910
	1024	128,752	15,045	8,558
	4096	526,962	44,115	11,945
4	64	10,356	3,218	3,218
	256	41,146	6,871	5,988
	1024	163,957	30,653	5,349
	4096	663,034	56,275	11,782

Table 2 shows the time of the algorithm and the resulting acceleration depending on the size of the swarm for Rastrigin function.

To illustrate the data presented in Table 1 and Table 2 in Figure 5 and Figure 6 shows a plot of running time on a single processor computer, and using CUDA technology for the test functions

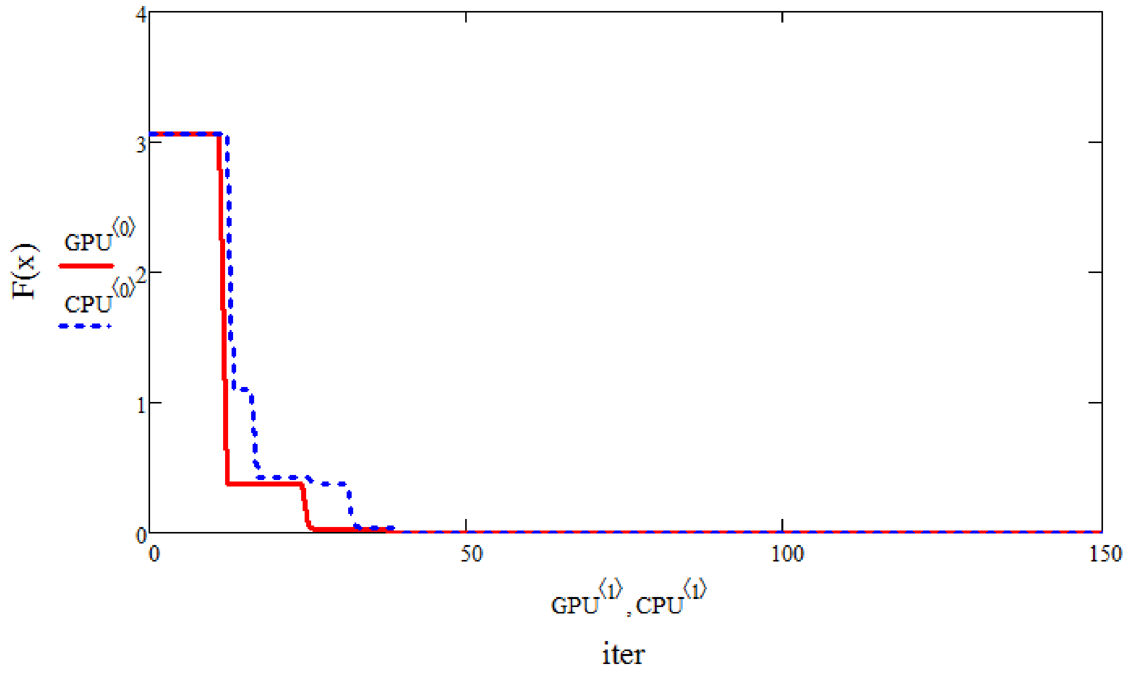

Fig5. *The rate of convergence of parallel and sequential particle swarm for spherical function*

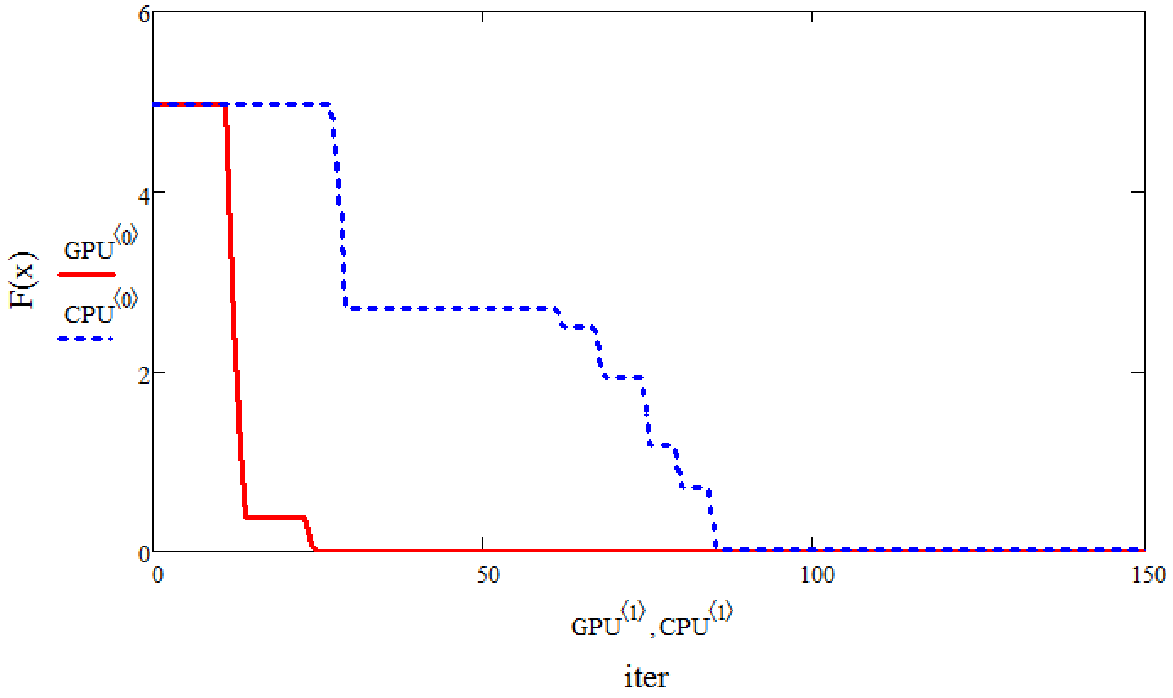

Fig6. *The rate of convergence of parallel and sequential particle swarm for the function Rastrigin*

III. Conclusion

As seen on the CPU time spent on finding an extremum more. It should be noted that the computational experiment on the GPU is also taken into account the time required to allocate memory on the graphics processor and then copy it to the original data.

Comparative analysis of the time serial and parallel algorithms, particle swarm method, showed that the use of CUDA technology reduces the solution of the problem (up to 12 times).

References

[1] D. Sanders, E. Kendrot CUDA by Example: An Introduction to General-Purpose GPU Programming. - DMK Press, 2011.

[2] The particle swarm algorithm - [electronic resource] - Access mode. - URL: http: //habrahabr.ru/post/105639/ (date accessed 03/02/2015)

[3] Y. Shi, R. Eberhart A modified particle swarm optimizer // The 1998 IEEE International Conference on Evolutionary Computation Proceedings, 1998

[4] Weise, T. Global Optimization Algorithms - Theory and Application: Ph.D. thesis / University of Kassel. - 2008.

[5] J Kennedy, R Eberhart Particle swarm optimization. // Proceedings of IEEE International conference on Neural Networks. - 1995.

[6] Barkalov KA Methods of parallel computing. Nizhny Novgorod: Publishing House of the Nizhny Novgorod State University them. NI Lobachevsky, 2011.

[7] Varygina MP Basics of programming in CUDA. Textbook. Krasnoyarsk: the Krasnoyarsk. state. ped. Univ them. VP Astaf'eva, 2012.

[8] Gergel VP High-performance computing for multi-core multi-processor systems. Textbook. Nizhny Novgorod: Publishing House of the UNN them. NI Lobachevsky, 2010.

[9] Boreskov AV, Kharlamov AA Technology basics CUDA. - M .: DMK Press, 2010.

[10] Official site of NVIDIA [electronic resource]. - Access mode: http://www.nvidia.ru/object/cuda-parallel-computing-ru.html (date accessed 03/02/2015)