Solving Time-Consuming Global Optimization Problems with Globalizer Software System

Alexander Sysoyev✉, Konstantin Barkalov, Vladislav Sovrasov, Ilya Lebedev, Victor Gergel

Lobachevsky State University of Nizhny Novgorod, Nizhny Novgorod, Russia
alexander.sysoyev@itmm.unn.ru
konstantin.barkalov@itmm.unn.ru
sovrasov.vlad@gmail.com
ilya.lebedev@itmm.unn.ru
victor.gergel@itmm.unn.ru

Abstract. In this paper, we describe the Globalizer software system for solving global optimization problems. The system implements an approach to solving the global optimization problems using the block multistage scheme of the dimension reduction, which combines the use of Peano curve type evolvents and the multistage reduction scheme. The scheme allows an efficient parallelization of the computations and increasing the number of processors employed in the parallel solving of the global optimization problems many times.

Keywords: Multidimensional Multiextremal Optimization · Global Search Algorithms · Parallel Computations · Dimension Reduction · Block Multistage Dimension Reduction Scheme

1 Introduction

The development of optimization methods that use high-performance computing systems to solve time-consuming global optimization problems is an area receiving extensive attention. The theoretical results obtained provide efficient solutions to many applied global optimization problems in various fields of scientific and technological applications. At the same time, the practical software implementation of these algorithms for multiextremal optimization is quite limited. Among the software for the global optimization, one can select the following systems:

- LGO (Lipschitz Global Optimization) [1] is designed to solve global optimization problems for which the criteria and constraints satisfy the Lipschitz condition. The system is a commercial product based on diagonal extensions of one-dimensional multiextremal optimization algorithms.
- GlobSol [2] is oriented towards solving global optimization problems as well as systems of nonlinear equations. The system includes interval methods based on the branch and bound method. There are some extensions of the system for parallel computations, and it is available to use for free.

- LINDO [3] is features by a wide spectrum of problem solving methods that can be used for these include linear, integer, stochastic, nonlinear, and global optimization problems. The ability to interact with the Microsoft Excel software environment is a key feature of the system. The system is widely used in practical applications and is available to use for free.
- IOSO (Indirect Optimization on the basis of Self-Organization) [4] is oriented toward solving of a wide class of the extremal problems including global optimization problems. The system is widely used to solve applied problems in various fields. There are versions of the system for parallel computational systems. The system is a commercial product, but is available for trial use.
- MATLAB Global Optimization Toolkit [5], includes a wide spectrum of methods for solving the global optimization problems, including multistart methods, global pattern search, simulated annealing methods, etc. The library is compatible to the TOMLAB system [6], which is an additional extension the widely-used MATLAB. It is also worth noting that similar libraries for solving global optimization problems are available for MathCAD, Mathematica, and Maple systems as well.
- BARON (Branch-And-Reduce Optimization Navigator) [7], is designed to solve continuous integer programming and global optimization problems using the branch and bound method. BARON is included in the GAMS (General Algebraic Modeling System) system used widely [8].
- Global Optimization Library in R [9] is a large collection of optimization methods implemented in the R language. Among these methods, there are stochastic and deterministic global optimization algorithms, the branch and bound method, etc.

The list provided above is certainly not exhaustive – additional information on software systems for a wider spectrum of optimization problems can be obtained, for example, in [10], [11], [12], etc. Nevertheless, even from such a short list the following conclusions can be drawn (see also [13]).

- The collection of available global optimization software systems for practical use is insufficient.
- The availability of numerous methods through these systems allows complex optimization problems to be solved in a number of cases, however, it requires a rather high level of user knowledge and understanding in the field of global optimization.
- The use of the parallel computing to increase the efficiency in solving complex time-consuming problems is limited, therefore, the computational potential of modern supercomputer systems is very poorly utilized.

In this paper, a novel Globalizer software system is considered. The development of the system was conducted based on the information-statistical theory of multiextremal optimization aimed at developing efficient parallel algorithms for global search – see, for example, [14–16]. The advantage of the Globalizer is that the system is designed to solve time-consuming multiextremal optimization problems. In order to obtain global optimized solutions within a reasonable time and cost, the system efficiently uses modern high-performance computer systems.

The paper is further structured as follows. In Section 2, the general statement of the multidimensional global optimization problem is considered. In Section 3, the Globalizer software system is presented and its architecture is described. In Section 4, the approaches to solving the multidimensional global optimization problem based on the information-statistical theory of multiextremal optimization is given. In Section 5, the results of applied problem solving with the Globalizer system are described. Finally, Section 6 presents the conclusion.

2 Statement of Multidimensional Global Optimization Problem

In this paper, the core class of optimization problems which can be solved using the Globalizer is examined. This involves multidimensional global optimization problems without constraints, which can be defined in the following way:

$$\varphi(y) \to \inf,\ y \in D \subset R^N, \tag{1}$$

$$D = \{y \in R^N:\ a_i \le y_i \le b_i, 1 \le i \le N\}, \tag{2}$$

i.e., a problem of finding the globally optimal values of the objective (minimized) function $\varphi(y)$ in a domain D defined by the coordinate bounds (2) on the choice of feasible points $y = (y_1,\ y_2, \dots, y_N)$.

If y^* is an exact solution of problem (1) – (2), the numerical solution of the problem is reduced to building an estimate y^0 of the exact solution matching to some notion of nearness to a point (for example, $\|y^* - y^0\| \le \varepsilon$ where $\varepsilon > 0$ is a predefined accuracy) based on a finite number k of computations of the optimized function values.

Regarding to the class of problems considered, the fulfillment of the following important conditions is supposed:

1. The optimized function $\varphi(y)$ can be defined by some algorithm for the computation of its values at the points of the domain D.
2. The computation of the function value at every point is a computation-costly operation.
3. Function $\varphi(y)$ satisfy the Lipschitz condition:

$$|\varphi(y_1) - \varphi(y_2)| \le L\|y_1 - y_2\|, \qquad \text{where } y_1, y_2 \in D, 0 < L < \infty, \tag{3}$$

that corresponds to a limited variation of the function value at limited variation of the argument.

The multiextremal optimization problems i.e. the problems, which the objective function $\varphi(y)$ has several local extrema in the feasible domain *D* in, are the subjects of consideration in the present paper. The dimensionality affects the difficulty of solving such problems considerably. For multiextremal problems so called "curse of dimensionality" consisting in an exponential increase of the computational costs with increasing dimensionality takes place.

3 Globalizer Architecture

The Globalizer considered in this paper expands the family of global optimization software systems successively developed by the authors during the past several years. One of the first developments was the SYMOP multiextremal optimization system [17], which has been successfully applied for solving many optimization problems. A special place is occupied by the ExaMin system [18], which was developed and used extensively to investigate the application of novel parallel algorithms to solve global optimization problems using high-performance multiprocessor computing systems.

The program architecture of Globalizer system is presented in Fig. 1.

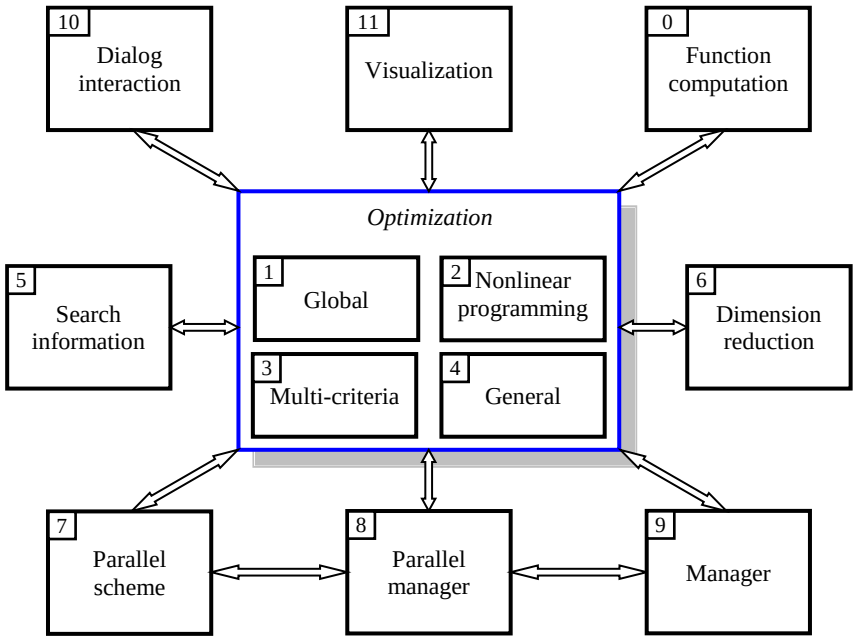

Fig. 1. Program architecture of Globalizer system (Blocks 1-2, 5-7 have been implemented; Blocks 3-4 and 8-11 are under development)

The structural components of the systems are:

- Block 0 is an external block. It consists of the procedures for computing the function values (criteria and constraints) for the optimization problem being solved.
- Blocks 1-4 form the optimization subsystem and solve the global optimization problems (Block 1), nonlinear programming (Block 2), multicriterial optimization (Block 3), and general decision making problems (Block 4). It is worth noting the successive scheme of interaction between these components – the decision making problems are solved using the multicriterial optimization block, which, in turn, uses the nonlinear programming block, etc.

- Block 5 is a subsystem for accumulating and processing the search information; this is one of the main subsystems – the amount of search information for time-consuming optimization problems may appear to be quite large on the one hand, but, on the other hand, the efficiency of the global optimization methods depends to a great extent on how completely all of the available search data is utilized.
- Block 6 contains the dimensional reduction procedures based on the Peano evolvents; this block also provides interaction between the optimization blocks and the initial multidimensional optimization problem.
- Block 7 organizes the choice of parallel computation schemes in the Globalizer system subject to the computing system architecture employed (the numbers of cores in the processors, the availability of shared and distributed memory, the availability of accelerators for computations, etc.) and the global optimization methods applied.
- Block 8 is responsible for managing the parallel processes when performing the global search (determining the optimal configuration of parallel processes, distributing the processes between computing elements, etc.).
- Block 9 is a management subsystem, which fully controls the whole computational process when solving global optimization problems.
- Block 10 is responsible for organizing the dialog interaction with users for stating the optimization problem, adjusting system parameters (if necessary), and visualizing and presenting the global search results.
- Block 11 is a set of tools for visualizing and presenting the global search results; the availability of tools for visually presenting the computational results enables the user to provide efficient control over the global optimization process.

4 Globalizer Approach for Solving the Global Optimization Problems

4.1 Methods of Dimension Reduction

Globalizer implements a block multistage scheme of dimension reduction [18], which reduces the solving of initial multidimensional optimization problem (1) – (2) to the solving of a sequence of «nested» problems of less dimensionality.

Thus, initial vector y is represented as a vector of the «aggregated» macro-variables

$$y = (y_1, y_2, \dots, y_N) = (u_1, u_2, \dots, u_M) \quad (4)$$

where the i-th macro-variable u_i is a vector of the dimensionality N_i from the components of vector y taken sequentially i. e.

$$\begin{gathered} u_1 = (y_1, y_2, \dots, y_{N_1}), \\ u_2 = (y_{N_1+1}, y_{N_1+2}, \dots, y_{N_1+N_2}), \dots \\ u_i = (y_{p+1}, \dots, y_{p+N_i}) \text{ where } p = \sum_{k=1}^{i-1} N_k, \dots \end{gathered} \quad (5)$$

at that, $\sum_{k=1}^{M} N_k = N$.

Using the macro-variables, the main relation of the well-known multistage scheme can be rewritten in the form

$$\min_{y\in D}\varphi(y)=\min_{u_1\in D_1}\min_{u_2\in D_2}\ldots\min_{u_M\in D_M}\varphi(y), \quad (6)$$

where the subdomains $D_i, 1\le i\le M$, are the projections of the initial search domain D onto the subspaces corresponding to the macro-variables $u_i, 1\le i\le M$.

The fact, that the nested subproblems

$$\varphi_i(u_1,\ldots,u_i)=\min_{u_{i+1}\in D_{i+1}}\varphi_{i+1}(u_1,\ldots,u_i,u_{i+1}), 1\le i\le M, \quad (7)$$

are the multidimensional ones in the block multistage scheme is the principal difference from the initial scheme. Thus, this approach can be combined with the reduction of the domain *D* (for example, with the evolvent based on Peano curve) for the possibility to use the efficient methods of solving the one-dimensional problems of the multiextremal optimization [19].

The Peano curve $y(x)$ lets map the interval of the real axis [0,1] onto the domain D uniquely:

$$\{y\in D\subset R^N\}=\{y(x): 0\le x\le 1\}. \quad (8)$$

The evolvent is the approximation to the Peano curve with the accuracy of the order 2^{-m} where m is the density of the evolvent.

Application the mappings of this kind allows reducing multidimensional problem (1) – (2) to a one-dimensional one

$$\varphi(y^*)=\varphi(y(x^*))=min\{\varphi(y(x)): x\epsilon[0,1]\}. \quad (9)$$

4.2 Method for Solving the Reduced Global Optimization Problems

The information-statistical theory of global search formulated in [14], [16] has served as a basis for the development of a large number of efficient multiextremal optimization methods – see, for example, [20–23], [24–27], etc. Within the framework of information-statistical theory, a general approach to parallelization computations when solving global optimization problems has been proposed – the parallelism of computations is provided by means of simultaneously computing the values of the minimized function $\varphi(y)$ at several different points within the search domain D – see, for example, [15], [16]. This approach provides parallelization for the most costly part of computations in the global search process.

Let us consider the general computation scheme of Parallel Multidimensional Algorithm of Global Search that is implemented in Globalizer.

Let us introduce a simpler notation for the problem being solved

$$f(x)=\varphi\big(y(x)\big): x\in[0,1]. \quad (10)$$

Let us assume $k>1$ iterations of the methods to be completed (the point of the first trial x^1 can be an arbitrary point of the interval [a; b] – for example, the middle of the interval). Then, at the $(k+1)$-th iteration, the next trial point is selected according to the following rules.

Rule 1. To renumber the points of the preceding trials $x^1,\ldots,x^n$ (including the boundary points of the interval [a; b]) by the lover indices in the order of increasing values of the coordinates,

$$0=x_0<x_1<\cdots<x_i<\cdots<x_k<x_{k+1}=1 \quad (11)$$

The function values $z_i = \varphi(x_i)$ have been calculated in all points $x_i, i = 1,..k$. In the points $x_0 = 0$ and $x_{k+1} = 1$ the function values has not been computed (these points are used for convenience of further explanation).

Rule 2. To compute the values:

$$\mu = \max_{1 \le i \le k} \frac{|z_i - z_{i-1}|}{\Delta_i}, \qquad M = \begin{cases} r\mu, & \mu > 0, \\ 1, & \mu = 0, \end{cases} \tag{12}$$

where $r > 1$ is the *reliability* parameter of the method, $\Delta_i = x_i - x_{i-1}$.

Rule 3. To compute the characteristics for all intervals $(x_{i-1}; x_i), 1 < i < k+1$, according to the formulae:

$$\begin{gathered} R(1) = 2\Delta_1 - 4\frac{z_1}{M}; \qquad R(k+1) = 2\Delta_{k+1} - 4\frac{z_k}{M}; \\ R(i) = \Delta_i + \frac{(z_i - z_{i-1})^2}{M^2 \Delta_i} - 2\frac{z_i + z_{i-1}}{M}, 1 < i < k+1. \end{gathered} \tag{13}$$

Rule 4. To arrange the characteristics of the intervals obtained according to (13) in decreasing order

$$R(t_1) \ge R(t_2) \ge \cdots \ge R(t_k) \ge R(t_{k+1}) \tag{14}$$

and to select p intervals with the highest values of characteristics (p is the number of processors/cores used for the parallel computations).

Rule 5. To execute new trials at the points

$$x_{k+j} = \begin{cases} \dfrac{x_{t_j} + x_{t_j - 1}}{2}, t_j \in \{1, k+1\}, \\ \dfrac{x_t + x_{t_j - 1}}{2} - \operatorname{sign}(z_{t_j} - z_{t_j - 1}) \dfrac{1}{2r} \left[\dfrac{|z_{t_j} - z_{t_j - 1}|}{M} \right]^N, 1 < t_j < k+1. \end{cases} \tag{15}$$

4.3 Implementation of Parallel Algorithm of Global Optimization

Let us consider a parallel implementation of the block multistage dimension reduction scheme described in Subsection 4.1.

For the description of the parallelism in the multistage scheme, let us introduce a vector of parallelization degrees

$$\pi = (\pi_1, \pi_2, \ldots, \pi_M), \tag{16}$$

where π_i, $1 \le i \le M$, is the number of the subproblems of the $(i+1)$-th nesting level being solved in parallel, arising as a result of execution of the parallel iterations at the i-th level. For the macro-variable u_i, the number π_i means the number of parallel trials in the course of minimization of the function $\varphi_M(u_1, \ldots, u_M) = \varphi(y_1, \ldots, y_N)$ with respect to u_i at fixed values of $u_1, u_2, \ldots, u_{i-1}$, i.e. the number of the values of the objective function $\varphi(y)$ computed in parallel.

In the general case, the quantities $\pi_i, 1 \le i \le M$ can depend on various parameters and can vary in the course of optimization, but we will limit ourselves to the case when all components of the vector π are constant.

Thus, a tree of MPI-processes is built in the course of solving the problem. At every nesting level (every level of the tree) PMAGS is used. Let us remind that the parallelization is implemented by selection not a single point for the next trial (as in the

serial version) but p points, which are placed into p intervals with the highest characteristics. Therefore, if p processors are available, p trials can be executed in these points in parallel. At that, the solving of the problem at the i-th level of the tree generates the subproblems for the $(i+1)$-th level. This approach corresponds to such a method of organization of the parallel computations as a «master-slave» scheme.

When launching the software, the user specifies:

- A number of levels of subdivision of the initial problem (in other words, the number of levels in the tree of processes) M;
- A number of variables (dimensions) at each level ($\sum_{k=1}^{M} N_k = N$ where N is the dimensionality of the problem);
- A number of the MPI-processes and the distribution of these ones among the levels ($\pi = (\pi_1, \pi_2, \dots, \pi_M)$).

Let us consider an example:

$N = 10,\ M = 3,\ N_1 = 3,\ N_2 = 4,\ N_3 = 3,\ \pi = (2,\ 3,\ 0)$.

Therefore, we have 9 MPI-processes, which are arranged into a tree (Fig. 2: at every function φ_i varied parameters are shown only, the fixed values are not shown in the figure). According to N_1, N_2, N_3 we have the following macro-variables: $u_1 = (y_1, y_2, y_3)$, $u_2 = (y_4, y_5, y_6, y_7)$, $u_3 = (y_8, y_9, y_{10})$. Each node solves a problem from relation (10). The root (level #0) solves the problem with respect to the first N_1 variables of the initial N-dimensional problem. The iteration generates a problem of the next level at any point. The nodes of level #1 solve the problems with respect to N_2 variables with the fixed values of the first N_1 variables, etc.

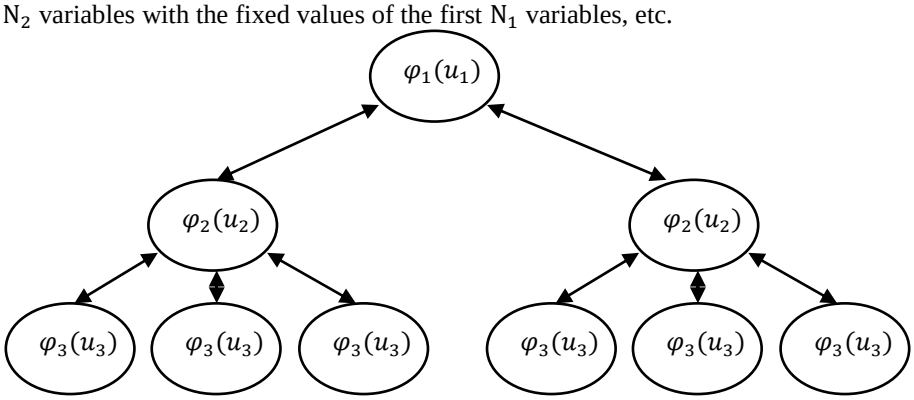

Fig. 2. Scheme of organization of parallel computations

5 Numerical Results

5.1 Test Problems Solving

The computational experiments were conducted using the Lobachevsky supercomputer at the State University of Nizhny Novgorod (http://hpc-education.unn.ru/en/

resources). The problems generated by the GKLS-generator [28] were selected for the test problems.

The results of the numerical experiments with Globalizer on an Intel Xeon Phi are provided in Table 1. The computations were performed using the Simple and Hard function classes with the dimensions equal to 4 and 5.

In the first series of experiments, serial computations using MAGS were executed. The average number of iterations performed by the method for solving a series of problems for each of these classes is shown in row I. The symbol ">" reflects the situation where not all problems of a given class were solved by a given method. It means that the algorithm was stopped once the maximum allowable number of iterations K_{max} was achieved. In this case, the K_{max} value was used for calculating the average number of iterations corresponding to the lower estimate of this average value. The number of unsolved problems is specified in brackets.

In the second series of experiments, parallel computations were executed on a CPU. The relative "speedup" in iterations achieved is shown in row II; the speedup of parallel computations was measured in relation to the serial computations (p = 1).

The final series of experiments was executed using a Xeon Phi. The results of these computations are shown in row III; in this case, the speedup factor is calculated in relation to the PMAGS results on a CPU using eight cores (p = 8).

Table 1. Average number of iterations

		p	N = 4		N = 5	
			Simple	*Hard*	*Simple*	*Hard*
I	**Serial computations** *Average number of iterations*	1	11953	25263	15920	>148342(4)
II	**Parallel computations on CPU** *Speedup*	2	2.51	2.26	1.19	1.36
		4	5.04	4.23	3.06	2.86
		8	8.58	8.79	4.22	6.56
III	**Parallel computations on Xeon Phi** *Speedup*	60	8.13	7.32	9.87	6.55
		120	16.33	15.82	15.15	17.31
		240	33.07	27.79	38.80	59.31

5.2 The Problem of Optimal Vibration Isolation for the Multi-degree-of-freedom System

Consider the vibration isolation problem for a multidegree-of-freedom system consisting of a base and elastic body to be isolated modeled by two material points connected each other by elastic and damping elements [29]. This mechanical system is described by the equations

$$\begin{aligned} &\ddot{\xi}_1 = -\beta\left(\dot{\xi}_1 - \dot{\xi}_2\right) - \xi_1 + \xi_2 + u + v, \\ &\ddot{\xi}_2 = -\beta\left(\dot{\xi}_2 - \dot{\xi}_1\right) - \xi_2 + \xi_1 + v, \\ &\xi_1(0) = \xi_2(0) = 0, \;\; \dot{\xi}_1(0) = \dot{\xi}_2(0) = 0. \end{aligned} \tag{17}$$

where ξ_1 and ξ_2 are coordinates of the material points, v is the base acceleration up to sign (the external excitation), u is the control force, β is a positive damping parameter. Rewrite the equation (26) in the standard form

$$\begin{aligned} \dot{x_1} &= x_3, \\ \dot{x_2} &= x_4, \\ \dot{x_3} &= -x_1 + x_2 - \beta x_3 + \beta x_4 + v + u, \\ \dot{x_4} &= x_1 - x_2 + \beta x_3 - \beta x_4 + v, \\ x_1(0) &= x_2(0) = x_3(0) = x_4(0) = 0. \end{aligned} \tag{18}$$

This model can describe the typical situations of vibration isolation for devices, apparatuses and humans located on moving vehicles.

Choose two criteria for this system to characterize the process of vibration isolation

$$J_1(u) = \sup_{v \in L_2} \frac{\sup_{t \geq 0} |x_1(t)|}{\|v\|_2}, \quad J_2(u) = \sup_{v \in L_2} \frac{\sup_{t \geq 0} |x_2(t) - x_1(t)|}{\|v\|_2}. \tag{19}$$

The first criterion characterizes the maximal displacement of the body to be isolated with respect to the base, while the second one the maximal deformation of the elastic body. Consider two-objective control problem for state-feedback case. The Pareto optimal front computed by Globalizer is presented on Fig. 3.

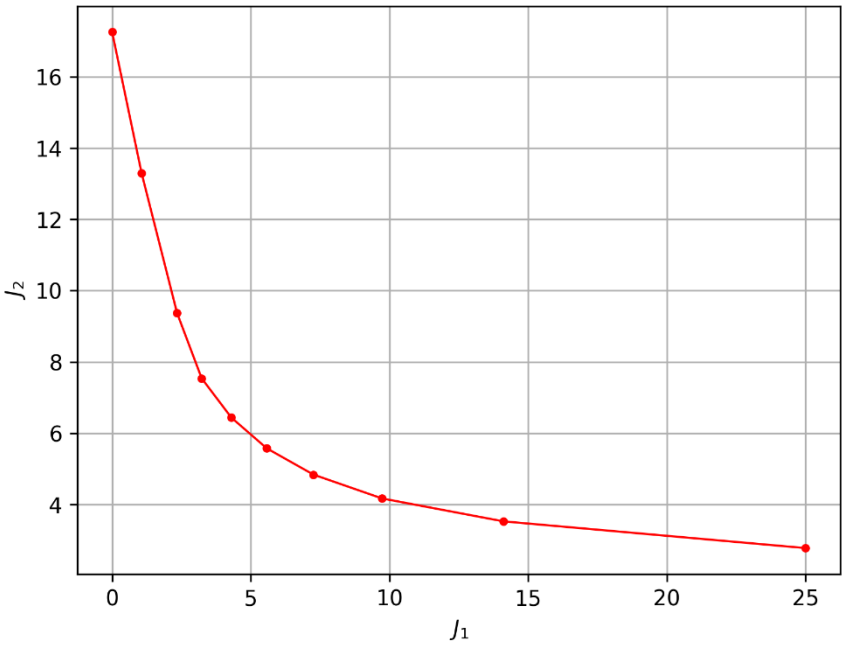

Fig. 3. Pareto optimal front for the vibration isolation problem

6 Conclusion

In this paper, the Globalizer global optimization software system was presented for implementing a general scheme for the parallel solution of globally optimized decision making. The work is devoted to the investigation of the possibility to speedup the process of searching the global optimum when solving the multidimensional multiextremal optimization problems using the approach based on the application of the parallel block multistage scheme of the dimension reduction.

The architecture of Globalizer system has been considered. The usage of Globalizer has been demonstrated by solving the applied problem of control theory.

7 Acknowledgements

This research was supported by the Russian Science Foundation, project No 16-11-10150 “Novel efficient methods and software tools for the time consuming decision making problems with using supercomputers of superior performance”.

References

1. Pintér, J.D.: Global Optimization in Action (Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications). Dordrecht: Kluwer Academic Publishers (1996).
2. Kearfott, R.B.: Globsol user guide. In: Optimization Methods and Software, 24, pp. 687-708 (2009).
3. Lin, Y., Schrage. L.: The global solver in the LINDO API. In: Optimization Methods and Software, pp. 657-668 (2009).
4. Egorov, I.N., Kretinin, G.V., Leshchenko, I.A. and Kuptzov, S.V.: IOSO Optimization Toolkit - Novel Software to Create Better Design. In: 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, Georgia (2002).
5. Venkataraman, P.: Applied Optimization with MATLAB Programming. John Wiley & Sons (2009).
6. Holmström, K., Edvall, M.M.: The TOMLAB Optimization Environment. In: Modeling Languages in Mathematical Optimization. Springer, 88, pp. 369-376 (2004).
7. Sahinidis, N.V.: BARON: A general purpose global optimization software package. Journal of global optimization 8 (2), pp. 201-205 (1996).
8. Bussieck, M.R., Meeraus, A.: Modeling Languages in Mathematical Optimization, chap. General Algebraic Modeling System (GAMS), pp. 137-157. Springer (2004).
9. Mullen, K.M.: Continuous Global Optimization in R. Journal of Statistical Software, 60 (6) (2014).
10. Mongeau, M., Karsenty, H., Rouz?, V., Hiriart-Urruty, J.B.: Comparison of public-domain software for black box global optimization. In: Optimization Methods and Software, 13 (3), pp. 203-226 (2000).
11. Pintér, J.D.: Software Development for Global Optimization. In: Pardalos, P.M. and Coleman, T.F. (eds.), Lectures on Global Optimization. Fields Institute Communications, 55, pp. 183-204 (2009).

12. Rios, L.M., Sahinidis, N.V.: Derivative-free optimization: a review of algorithms and comparison of software implementations. In: Journal of Global Optimization 56 (3), pp. 1247-1293 (2013).
13. Liberti, L.: Writing Global Optimization Software. In: Global Optimization: From Theory to Implementation. Springer, 84, pp. 211-262 (2006).
14. Strongin, R.G.: Numerical Methods in Multiextremal Problems (Information-Statistical Algorithms). Moscow: Nauka (In Russian) (1978).
15. Strongin, R.G., Gergel, V.P., Grishagin, V.A., Barkalov, K.A.: Parallel Computations for Global Optimization Problems. Moscow: Moscow State University (In Russian) (2013).
16. Strongin, R.G., Sergeyev, Ya.D.: Global Optimization with Non-convex Constraints: Sequential and Parallel Algorithms. Dordrecht, Kluwer Academic Publishers (2000).
17. Gergel, V.P.: A software system for multiextremal optimization. In: European Journal of Operational Research, 65 (3), pp. 305-313 (1993).
18. Barkalov K.A., Gergel V.P.: Multilevel scheme of dimensionality reduction for parallel global search algorithms. In: Proceedings of the 1st International Conference on Engineering and Applied Sciences Optimization, pp. 2111-2124 (2014).
19. Sergeyev, Ya.D., Strongin, R.G., Lera, D.: Introduction to Global Optimization Exploiting Space-Filling Curves. Springer (2013).
20. Barkalov, K., Gergel, V.: Parallel Global Optimization on GPU. In: Journal of Global Optimization, 66(1), 3-20 (2016)
21. Barkalov, K., Gergel, V., Lebedev, I.: Use of Xeon Phi Coprocessor for Solving Global Optimization Problems. LNCS, Springer, Heidelberg. 9251, 307-318 (2015).
22. Gergel, V.P., Grishagin, V.A., Gergel, A.V.: Adaptive nested optimization scheme for multidimensional global search. In: Journal of Global Optimization, 66(1), pp. 1-17 (2015)
23. Gergel, V., Lebedev, I.: Heterogeneous Parallel Computations for Solving Global Optimization Problems. In: Procedia Computer Science, 66, pp. 53-62 (2015).
24. Sergeyev Y.D.: An Information Global Optimization Algorithm with Local Tuning. In: SIAM Journal on Optimization, 5(4), pp. 858-870 (1995).
25. Sergeyev, Y.D.: Multidimensional Global Optimization Using the First Derivatives. In: Computational Mathematics and Mathematical Physics, 39(5), pp. 743-752 (1999).
26. Sergeyev, Y.D., Grishagin V.A.: A Parallel Method for Finding the Global Minimum of Univariate Functions. In: Journal of Optimization Theory and Applications, 80(3), pp. 513-536 (1994).
27. Sergeyev, Y.D., Grishagin, V.A.: Parallel Asynchronous Global Search and the Nested Optimization Scheme. In: Journal of Computational Analysis & Applications, 3(2), pp. 123-145 (2001).
28. Barkalov, K., Gergel, V., Lebedev, I.: Use of Xeon Phi Coprocessor for Solving Global Optimization Problems. LNCS, Springer, Heidelberg. 9251, 307-318 (2015).
29. Balandin, D.V., Kogan, M.M.: Pareto-optimal generalized H2-control and vibration isolation problems. Automation and Remote Control (2017) (article in press).