Journal of Computational Analysis and Applications, Vol. 3, No. 2, 2001

Parallel Asynchronous Global Search and the Nested Optimization Scheme

Yaroslav D. Sergeyev[1] and Vladimir A. Grishagin[2]

In this paper, a parallel asynchronous information algorithm for solving multidimensional Lipschitz global optimization problems, where times for evaluating the objective function can be different from point to point, is proposed. This method uses the nested optimization scheme and a new parallel asynchronous global optimization method for solving core univariate subproblems generated by the nested scheme. The properties of the scheme related to parallel computations are investigated. Global convergence conditions for the new method and theoretical conditions of speed up, which can be reached by using asynchronous parallelization in comparison with the pure sequential case, are established. Numerical experiments comparing sequential, synchronous, and asynchronous algorithms are also reported.

KEY WORDS: Global optimization; parallel asynchronous computations; nested optimization; convergence; speed up.

1. INTRODUCTION

In this paper, we consider the one-dimensional global optimization problem

$$\min\{f(y) : y \in [a, b]\} \tag{1}$$

and its multidimensional generalization

$$\min\{f(y) : y \in D\}, \tag{2}$$

$$D = \{y \in R^N : a_i \le y_i \le b_i, 1 \le i \le N\}. \tag{3}$$

It is supposed that $f(y)$ satisfies the Lipschitz condition with a constant $0 < K < \infty$. Many algorithms have been proposed to solve the problems

[1] ISI-CNR, c/o DEIS, University of Calabria, Rende (CS), 87036, Italy and Software Department, University of Nizhni Novgorod, Gagarin Ave. 23, Nizhni Novgorod, 603600, Russia. email: yaro@si.deis.unical.it

[2] Software Department, University of Nizhni Novgorod, Gagarin Ave. 23, Nizhni Novgorod, 603600, Russia. Email: vagris@unn.ac.ru

1521-1398/01/0400-0123$19.50/0

(1–3) (see, for example, Aluffi-Pentini, Parisi, and Zirilli [1]; Archetti and Schoen [2]; Baritompa [3]; Bomze *et al*. [6]; Dixon and Szego [8]; Grishagin [11]; Floudas and Pardalos [9]; Hansen and Jaumard [13]; Hendrix [14]; Horst and Pardalos [15]; Horst and Tuy [16]; Kearfott [17]; Pardalos and Rosen [22]; Pintér [25], [26]; Ratschek and Rokne [27]; Rinnooy Kan and Timmer [28]; Strongin [34]; Törn and Žilinskas [37]; Wood [38]; Zabinsky and Smith [39]; Zhigljavsky [40]; and references given therein).

In general, in real-life applications it is very difficult to deal with such problems. Not only the number of trials [hereinafter *trial* is evaluation of the objective function $f(y)$ at a point y] to obtain an ε approximation of the exact solution is very high, but execution of every trial also requires a lot of time. Parallel computations permit significant acceleration of the search (see Bertsekas and Tsitsiklis [5]). For this reason, in the last few years, attention of many researchers was directed to the creation of parallel global optimization methods (see Bertocchi [4]; Migdalas, Pardalos, and Støroy [19]; pardalos [21]. Pardalos, Phillips, and Rosen [92]).

One of approaches—parallel characteristical algorithms (see Gergel and Sergeyev [10]; Grishagin, Sergeyev, and Strongin [12])—describes a class of parallel methods (see Sergeyev and Strongin [33]; Strongin and Sergeyev [36]; Sergeyev and Grishagin [31,32]), designed to solve problem (1). By applying Peano-type space-filling curves for reduction of dimension (this approach has been introduced in Strongin [34] and is discussed in Grishagin, Sergeyev, and Strongin [12]; Sergeyev [29,30]; Strongin [34,35]; Strongin and Sergeyev [36]), it is possible to also use these methods for solving problems (2 and 3). The methods work by executing a few trials (every trial is performed at a separate processor) in parallel.

The parallel characteristical methods are synchronous and, therefore, demonstrate good performance dealing with problems where the time of evaluating $f(y)$ is the same over the whole search region. If this condition is not satisfied, parallel processors are forced to stand idle, waiting until all processors involved in the parallel iteration complete their trials.

In the next section, we propose to use the nested optimization scheme in order to construct a multidimensional parallel asynchronous algorithm for solving problems (2 and 3). This scheme was initially proposed in Carr and Howe [7] and Pijavskii [24] and is interesting to the authors for the following reasons: (1) it offers a quite simple way of generalizing many univariate global optimization methods to the multidimensional case and so can be easily used by engineers in solving their applied problems; (2) while working, it generates a series of univariate subproblems, which can be efficiently solved in parallel; (3) because of the nature of the nested optimization scheme, almost all generated subproblems have different trial times at different points of the search region, even in the case when the initial problems

(2 and 3) do not have this property. Thus, the role of asynchronous calculations is decisive (synchronous calculations cannot be efficiently applied because of idle times of parallel processors).

The rest of the paper is structured as follows. Section 3 presents a new parallel asynchronous information algorithm for solving problems, where trial times can be different from point to point. This method is used as the core algorithm for solving the univariate subproblems of the type (1) generated by the nested optimization scheme. Section 4 contains convergence conditions of the method. It also presents theoretical conditions of speed up, which can be obtained in comparison with the sequential case because of parallel asynchronous computations. Section 5 describes numerical experiments, where the purely sequential, synchronous, and asynchronous methods are compared. Finally, Section 6 concludes the paper.

2. NESTED OPTIMIZATION SCHEME AND PARALLEL COMPUTATIONS

The scheme of nested optimization is based on the known relation (see, e.g., Carr and Howe [7] and Pijavskii [24]):

$$\min_{y\in D} f(y) = \min_{a_1\le y_1\le b_1} \min_{a_2\le y_2\le b_2} \dots \min_{a_N\le y_N\le b_N} f(y) \tag{4}$$

which permits the multidimensional problems (2 and 3) to be solved via a set of univariate problems and, therefore, to apply the methods for the minimization of one-dimensional functions to investigate the multivariate problems.

Let us introduce a set of functions defined recursively as

$$f_N(y_1, \dots, y_N) = f(y_1, \dots, y_N), \tag{5}$$

$$f_i(y_1, \dots, y_i) = \min_{a_{i+1}\le y_{i+1}\le b_{i+1}} f_{i+1}(y_1, \dots, y_i, y_{i+1}), \qquad 1 \le i \le N. \tag{6}$$

Then, in accordance with (4), in order to find the solution to problems (2 and 3), it is required to solve the univariate problem

$$f_1(y_1) \Rightarrow \min, \qquad y_1 \in [a_1, b_1]. \tag{7}$$

However, each evaluation of $f(y_1)$ at a fixed point y_1 requires solving the univariate problem

$$f_2(y_1, y_2) \Rightarrow \min, \qquad y_2 \in [a_2, b_2] \tag{8}$$

and so on, up to the minimization of $f_N(y) = f(y)$.

In the general scheme, we have to solve the set of "nested" univariate optimization problems

$$f_i(y_1, \ldots, y_i) \Rightarrow \min, \qquad y_i \in [a_i, b_i] \tag{9}$$

for $1 \le i \le N$, where the number i denotes the level of recursion in (6), or just "the level of optimization." The first level (7) is supposed to be the highest one. Note, that in (9) the coordinates y_j, $1 \le j \le i-1$, are fixed [these coordinates are given by the preceding higher-level problems (9)].

The scheme (5–9) in combination with sequential univariate optimization algorithms can be used as the basis of designing a number of sequential multidimensional methods (see Carr and Howe [7]; Pijavskii [24]; and Strongin [34]). This approach can be also used for elaborating multivariate parallel algorithms by applying parallel univariate methods for solving the problems of the type (9).

Let us discuss this possibility in detail. The inclusion of parallel univariate methods into the nested optimization scheme leads to solving several problems (9) in parallel. Indeed, a sequential method, in solving the problem (7), requires the computation of one value of the function $f(y_1)$ in the course of every iteration and, therefore, generates one problem (8) only. In contrast with this, a parallel method used for the same goal instead of evaluating the objective function at the only point computes its values simultaneously at several points and, thus, several problems of the type (8) are to be solved in parallel. Similar reasoning holds in the general case (9), as well, when $1 \le i \le N-1$.

For the regulation of the process, which generates the parallel subproblems, we introduce the vector of parallelization degrees

$$q = (q_1, q_2, \ldots, q_N), \tag{10}$$

where q_i, $1 \le i \le N-1$, is the number of parallel univariate minimization subproblems of the $(i+1)$th recursion level arising as the result of the execution of a parallel iteration at the ith recursion level. For the coordinate y_N, the number q_N is the number of parallel trials during the minimization of the function $f_N(y) = f(y_1, y_2, \ldots, y_N)$ with fixed values of $y_1, \ldots, y_{N-1}$, i.e., for $i = N$ in (9), a parallel iteration of the optimization method consists of the parallel computation of the values of the objective function $f(y)$, rather than the parallel handling of the subproblems of the lower level.

In the general case, the values q_i, $1 \le i \le N$, can depend on some parameters, for example, on the coordinates $y_1, \ldots, y_{i-1}$ (if $i \ge 2$): $q_i = q_i(y_1, \ldots, y_{i-1})$, but further, for simplicity, the components q_i, $1 \le i \le N$, are considered to be constant, i.e., every optimization level is characterized by the constant degree of parallelization.

The use of a parallel algorithm in combination with scheme (5–9) and vector (10) permits, for solving the problems (2 and 3), up to Q processors functioning in parallel, where

$$Q = \prod_{i=1}^{N} q_i. \tag{11}$$

In this case, the nested optimization scheme generates up to Q/q_N univariate global minimization subproblems to be solved simultaneously.

So, for example, if $N = 2$ then the use of a method with q_1 parallel trials in iteration for solving the external univariate minimization problem (7) generates q_1 internal one-dimensional problems (8) and each can be solved by a parallel method with q_2 trials in iteration. This means that at the most internal level, corresponding to $i = N$ in (9), $Q = q_1 q_2$ values of the function $f(y)$ are calculated, i.e., for the computation of these values we can use Q processors working in parallel. In Fig. 1, we present two possible distributions of $Q = 6$ parallel processors.

If we combine the nested scheme with a synchronous algorithm, in which the stopping rule differs from the execution of the fixed number of iterations, then inevitably we get a situation when a group of processors will stand idle. Indeed, in accordance with (10), during the univariate minimization of the ith level (9), the parallel iteration consists of solving q_i minimization problems of the $(i+1)$th level, $1 \le i \le N-1$. Even in that case, when the time of the objective function $f(y)$ evaluation is the same at all points of the feasible region D, the number of trials executed before termination of the univariate search is different for every jth subproblem of $(i+1)$th level, $1 \le j \le q_i$. If, besides, the computation time of $f(y)$ is not identical for different points $y \in D$, then the times for solving the subproblems can differ even more. It means that

$$\prod_{j=1+1}^{N} q_j$$

processors which have already solved their subproblems will wait until the last of q_ith subproblems will be solved.

These idle times can be eliminated if, during the minimization at the ith level, the group of processors that have solved their subproblem of $(i+1)$th level will not wait for the completion of the search by all

$$q_i \cdot \prod_{j=i+1}^{N} q_j$$

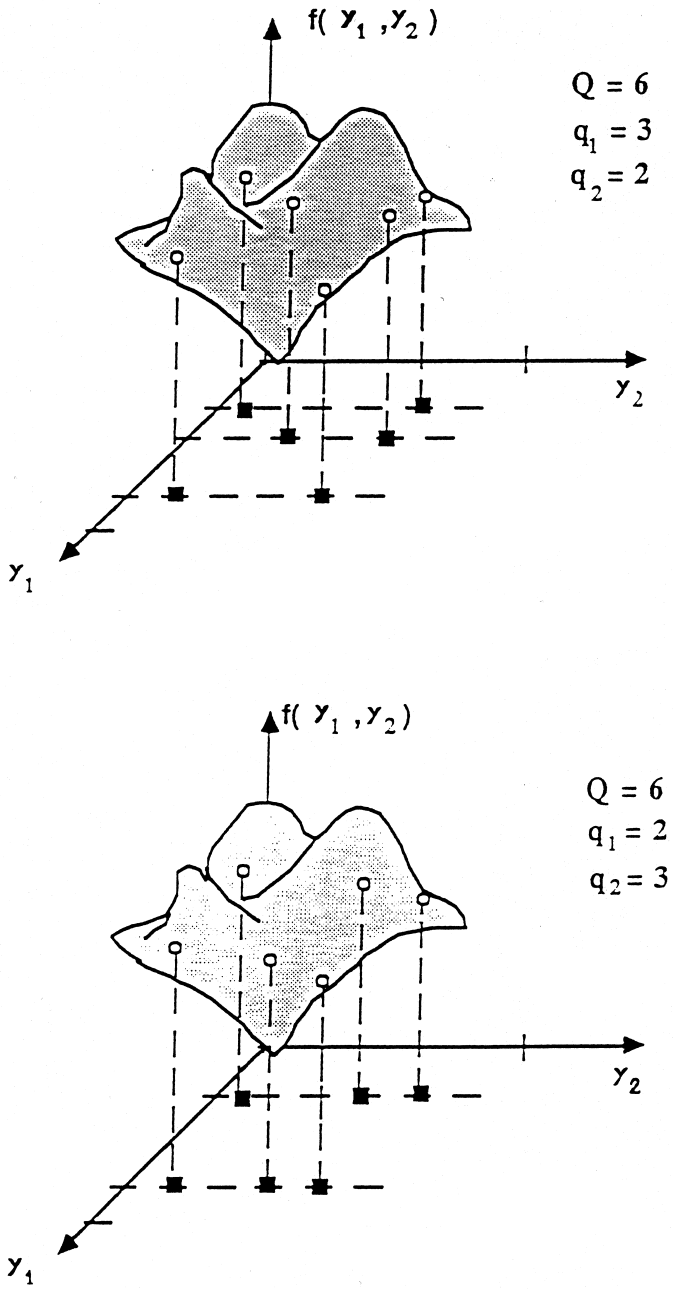

Fig. 1.

processors and will begin to solve a new subproblem of type (6) immediately. Thus, the parallel iterations, in the sense of synchronous algorithms, disappear and the search control is asynchronous. However, the implementation of such a possibility requires the creation of special parallel asynchronous algorithms for the univariate global optimization. We pursue exactly this object in the next section where a new algorithm with the asynchronous decision rule is proposed.

3. THE CORE ASYNCHRONOUS PARALLEL ALGORITHM FOR UNIVARIATE GLOBAL OPTIMIZATION PROBLEMS

Let us rewrite problem (9) of the nested optimization scheme in a simplified standard form

$$\phi(x) \Rightarrow \min, \qquad x \in [a, b] \tag{12}$$

i.e., as the usual problem of univariate optimization over the closed interval.

The asynchronous parallel algorithm (APA) with p parallel processors for solving the problem (12) in the case when the times of evaluation of the function $\phi(x)$ can differ for different points $x \in [a, b]$ consists of the following.

First $K \geq p + 1$ trials are performed at the points $x^1 = a$, $x^2 = b$, and at arbitrary internal points $x^3, x^4, \ldots, x^K$ of the interval $[a, b]$. In this stage, if $p < K$, a processor having completed its trial at a point x^i, $1 \leq i < K$ [i.e., having evaluated the function $\phi(x)$ at the point x^i] does not wait for termination of the work of other processors, but immediately starts to perform the new trial at a new point x^{k+p}, if $k < K$, where k is the number of completed trials.

After the initial stage, each processor that completed its trial begins immediately to choose a new trial point x^{k+p} ($k \geq K$) according to the basic decision rule of the algorithm described below. In this situation, we have the set

$$I_k = \{x^{k+1}, x^{k+2}, \ldots, x^{k+p-2}, x^{k+p-1}\}$$

of the trial points, where the trials have not yet been completed.

Step 1. The points of the set

$$X_k = \{x^1, \ldots, x^{k+p-1}\} \tag{13}$$

including the points x^j, $1 \le j \le k+p-1$, of preceding trials are renumbered (by subscripts) in the order of increasing coordinates, namely

$$a = x_1 < x_2 < \cdots < x_{k+p-1} = b. \tag{14}$$

Step 2. The value

$$\mu = \max\{M_1, M_2\}, \tag{15}$$

where

$$M_1 = \max\{|z_i - z_{i-1}|/(x_i - x_{i-1}) : x_{i-1} \notin I_k, x_i \notin I_k, 2 \le i \le k+p-1\},$$

$$M_2 = \max\{|z_{i+1} - z_{i-1}|/(x_{i+1} - x_{i-1}) : x_i \in I_k, 2 \le i < k+p-1\},$$

is computed where $z_i = \phi(x_i)$, if $x_i \notin I_k$, $1 \le i \le k+p-1$. The values z_i, such that $x_i \in I_k$ are not defined (the trials at the points $x_i \in I_k$ have not yet been completed). If the value μ cannot be computed, it is considered to be equal to one.

Step 3. The real number

$$R(i) = r\mu\Delta_i + \frac{(z_i - z_{i-1})^2}{r\mu\Delta_i} - 2(z_i + z_{i-1}), \tag{16}$$

$$\Delta_i = x_i - x_{i-1},$$

is assigned to each subinterval $[x_{i-1}, x_i]$, $2 \le i \le k+p-1$. The value $R(i)$ is called the *characteristic* of the interval $[x_{i-1}, x_i]$ and $r > 1$ is the reliability parameter of the method.

Step 4. An interval $[x_{t-1}, x_t]$, to which the highest characteristic $R(t)$ corresponds, i.e.,

$$R(t) = \max\{R(i) : X_{i-1} \notin I, x_i \notin I, 2 \le i \le k+p-1\} \tag{17}$$

is chosen.

Step 5. The point of the next trial is defined as

$$x^{k+p} = 0.5(x_t + x_{t-1}) - (z_t - z_{t-1})/(2r\mu), \tag{18}$$

where t is from (17).

The computational scheme of the algorithm has been described.

Remark 1. The algorithm described above belongs to the class of information global optimization algorithms introduced in Strongin [34] (see also Strongin [35]; Strongin and Sergeyev [36]; Sergeyev and Strongin [33]; Sergeyev [29]). The normalized characteristics $R(i)$ from (16) can be viewed as probabilities of the locating of a global minimizer within the interval $[x_{i-1}, x_i]$, $1 \le i \le k+p-1$. Thus, during every iteration, the new trial point is chosen within the interval having the highest probability to contain the global optimum. The purely sequential prototype (see Strongin [34,35]) can be extracted from the above scheme by taking the number of parallel processors $p = 1$.

Remark 2. The above scheme does not contain any *stopping rule*, i.e., it is supposed to generate an infinite sequence $\{x^k\}$ of search trials (provided the time of computation for each value z^k is finite). The problem of stopping rule will be discussed later after establishing theoretical results concerning the convergence conditions.

Remark 3. It can be shown (see Pijavskii [24]; Strongin [34]) that the formula (18) corresponds to the global minimizer of the following piecewise linear function

$$h(x) = \max\{\phi(x_{t-1}) - r\mu|x - x_{t-1}|, \phi(x_t) - r\mu|x_t - x|\}$$
$$x \in [x_{t-1}, x_t],$$

with the properties

$$h(x_{t-1}) = \phi(x_{t-1}), \qquad h(x_t) = \phi(x_t), \qquad h(x) \le \phi(x), \qquad x \in [x_{t-1}, x_t].$$

Thus, x^{k+p} is the point corresponding to the minimal possible value over the interval $[x_{t-1}, x_t]$ of any function having the Lipschitz constant equal to $r\mu$.

4. CONVERGENCE AND NONREDUNDANCY CONDITIONS

Consider a trial sequence $\{x^k\}$ generated by the APA while solving problem (12). With this sequence, we can connect p sequences $\{T_s^j\}$, $1 \le j \le p$, where T_s^j is the moment of completion of the sth trial executed by the jth processor. Since, in accordance with (13–18), any processor having completed a trial begins immediately to compute the value $\phi(x)$ at a new point, the time of computation of the sth trial by the jth processor can be defined as the difference

$$T_s^j - T_{s-1}^j, \qquad 1 \le j \le p, s > 1.$$

Theorem 1. Let during the minimization by the APA of a function $\phi(x)$, $x \in [a, b]$, satisfying the Lipschitz condition with a constant K, the number p of processors cannot exceed a constant Q, i.e., $1 < p \leq Q < \infty$, and for the moments of trial completions the relation

$$T^j_s - T^j_{s-1} < T < \infty, \qquad 1 \leq j \leq p, \qquad s > 1 \tag{19}$$

be true. Then the convergence to any limit point $\tilde{x}$ of the trial sequence $\{x^k\}$ generated by the APA will be bilateral (double-sided), if $\tilde{x}$ is an inner point of the interval $[a, b]$.

Proof. This theorem can be proved analogously to the proof of the similar theorem for the synchronous information method given in Sergeyev and Strongin [33] and Strongin and Sergeyev [36]. What remains to be done is the taking into account the assumption of asynchronism. In fact, condition (19) ensures the impossibility of a situation when a processor having begun a trial cannot complete it. □

Bilateral convergence stated by the Theorem 1 permits the introduction into the APA computational scheme, a stopping rule of the type

$$x_t - x_{t-1} \leq \varepsilon, \tag{20}$$

where t is from (17 and 18) and $\varepsilon > 0$ is the search tolerance. Thus, according to (20), the APA stops running if the length of the interval with the highest characteristic becomes less than the given accuracy ε. Note, that in the case $\tilde{x} = a$ or $\tilde{x} = b$ the unilateral convergence is sufficient for the fulfillment of (20).

Theorem 2. Under the conditions of Theorem 1, the following assertions are true: (i) any limit point $\tilde{x}$ is locally optimal if the objective function $\phi(x)$ has a finite number of local extrema; (ii) if there exists another limit point $\bar{x} \neq \tilde{x}$, then it follows $\phi(\bar{x}) = \phi(\tilde{x})$; (iii) the values $\phi(x^k) \geq \phi(\tilde{x})$ for any $k \geq 1$; (iv) if there exists an iteration number m^* such that for all steps $m > m^*$ the product $r\mu$ from (16 and 18) satisfies the inequality

$$r\mu \geq K_j + \sqrt{K_j^2 - \mu_j^2}, \tag{21}$$

$$\mu_j = |z_j - z_{j-1}|/(x_j - x_{j-1}),$$

where K_j is the local Lipschitz constant of the interval $[x_{j-1}, x_j]$ containing a global minimizer x^* during the current iteration, i.e.,

$$x^* \in [x_{j-1}, x_j], \tag{22}$$

then x^* will be the limit point of the trial sequence generated by the APA.

Proof. The theorem states that the introduction of the asynchronous parallelism does not change the convergence properties inherent to the sequential prototype (see Strongin [36]). In particular, the APA does not produce limit points being different from those of the synchronous method from Sergeyev and Strongin [33]. The propositions (i–iii) are proved by repeating the proof of Theorem 2 from Sergeyev and Strongin [33]. The last proposition is proved by a complete analogy with the proof of Theorem 1.1 from Sergeyev [29]. □

Corollary 1. If condition (21) takes place, then only global minimizers of $\phi(x)$ can be limit points of the sequence $\{x^k\}$ generated by the APA.

Proof. This result follows immediately from assertions (ii) and (iv) of the theorem. □

Corollary 2. If there exists an iteration number $\tilde{m}$ such that for all steps $m > \tilde{m}$ condition (21) takes place for the intervals containing all the global minimizers then all these points will be the limit points of the sequence $\{x^k\}$.

Proof. The proof is obvious and is so omitted. □

Let us now discuss which levels of speed up in comparison with the purely sequential case can be reached by the APA. The parallelism introduced in the decision rules (13–18) can generate in its trial sequence some additional, or *redundant* points, with regard to the sequence of the purely sequential decision rule. Naturally, appearance of the redundant points can slow down the search and so reduce the efficiency of use of the parallel processors. This effect can take place for the following reason.

When the sequential method chooses the point x^{k+p} during its $(k+p)$th iteration, it has the complete information hitherto obtained during the previous $k+p-1$ iterations at the points $x^1, \ldots, x^{k+p-1}$. On the contrary, when the APA chooses the point x^{k+p}, it cannot use the results of the trials at the points x^{k+1}, $x^{k+2}, \ldots, x^{k+p-1}$ because they have not yet been completed. This lack of the search information can lead to generating redundant trial points which differ from those where the sequential method perform its trials.

In order to characterize the redundancy of the asynchronous parallelism, we introduce, following studies of parallel synchronous methods presented in Sergeyev and Strongin [33], Strongin and Sergeyev [36], Sergeyev and Grishagin [31,32], Grishagin, Sergeyev, and Strongin [12], a number of quantitative coefficients.

Let, while handling problem (12), a sequential method and its parallel generalization, produce the infinite sequences of trial points $\{w^j\}$ and $\{x^k\}$, respectively, using the decision rules without a stopping condition. If

$$\{w^j\} = \{x^k\} \tag{23}$$

then the parallel and sequential methods place trials at the same points [no matter whether the element-by-element coincidence occurs in (23)]. Case (23) corresponds to the *nonredundant* parallelization and provides the highest speed up.

When situation

$$\{w^j\} \subset \{x^k\}$$

takes place, it implies the redundancy of the parallel scheme.

Let us consider the value

$$\tau^*(m, n) = \operatorname{card}\left(\left\{x^{n+1}, \ldots, x^m\right\} \setminus \left\{w^j\right\}\right), \qquad m > n \geq 0 \tag{24}$$

being the number of redundant points in $\{x^k\}$ from the $(n+1)$th to the mth trial. By using (24), we can now define the *redundancy coefficient*

$$\tau(m, n) = \tau^*(m, n)/(m - n). \tag{25}$$

It is clear that $\tau(m, n) = 0$ means the nonredundancy of the trial series $\{x^{n+1}, \ldots, x^m\}$ of the parallel algorithm. Nonredundant parallelization (23) corresponds to the case $\tau(m, 0) = 0$ for all $m > 0$.

Let us now establish conditions of the nonredundant parallel asynchronous search for the APA. In the further theoretical analysis, we suppose that the objective function $\phi(x)$, $x \in [a, b]$, is Lipschitzian with the constant K, $0 < K < \infty$, and for the APA the following condition holds

$$w^s = x^s, \qquad 1 \leq s \leq K, \tag{26}$$

i.e., the initial trial points are the same for the sequential and parallel methods. We also suppose that for the value μ, a constant $M > 0$ is used for all $k > K$ (this assumption is not very restrictive because this situation takes place for sufficiently large numbers of the APA iterations).

Theorem 3. Let x^* be a global minimizer and x' a local one for a function $\phi(x)$, $x \in [a, b]$, and the inequality

$$\phi(x') - \phi(x^*) \leq \delta \tag{27}$$

be true for a number $\delta > 0$. If: (i) there exists a trial point $x^n \in \{x^k\}$ falling between x^* and x', i.e., $x' \leq x^n \leq x^*$ or $x^* \leq x^n \leq x'$; (ii) condition (19) takes place; (iii) there exists an iteration number m^*, such that for all steps $m > m^*$ the product $r\mu$ from (16) and (18) satisfies the inequality

$$r\mu > \max\left\{K_j + \sqrt{K_j^2 - \mu_j^2}, K_i + \sqrt{K_i^2 - \mu_i^2}\right\}, \qquad j \neq i, \tag{28}$$

where $K_j\mu_j$ correspond to the interval (22) and K_i is the local Lipschitz constant corresponding to the interval

$$[x_{i-1}, x_i] \ni x' \tag{29}$$

and

$$\mu_i = |z_i - z_{i-1}|/(x_i - x_{i-1}). \tag{30}$$

Then, the APA with two parallel processors provides $\tau(m, n) = 0$ while

$$x_t - x_{t-1} > \tilde{\varepsilon}, \tag{31}$$

where t is from (17) and

$$\tilde{\varepsilon} = \frac{4\delta}{r\mu + \mu_i^2(r\mu)^{-1} - 2K_i}. \tag{32}$$

Proof. If after the mth trial of the APA and the kth trial of the purely sequential method, we have the equality

$$(x_{t-1}, x_t) = (w_{j-1}, w_j), \qquad t = t(m), \qquad j = j(k), \tag{33}$$

and the current trials at the points x^{m+p} and w^{k+1} fall within these respective intervals, then

$$R(t(m)) = R(j(k))$$

and from (18) we obtain

$$x^{m+p} = w^{k+1}, \tag{34}$$

since μ is a constant and condition (28) is fulfilled.

Condition (28) and Corollary 1 imply that the only limit points of the sequences $\{w^k\}$ and $\{x^m\}$ are the global minimizers of the function $\phi(x)$ and, thus, the set of limit points of the sequences $\{x^m\}$ coincides with the set of limit points of the sequence $\{w^k\}$.

Let the first m points of the sequence $\{w^k\}$ be ordered by the scheme (14) and $j = j(k)$ be the number of the interval $[x_{j-1}, x_j]$ from (22). By condition (28), the characteristic of this interval satisfies the inequality

$$R(j(k)) > -4\phi(x^*) \tag{35}$$

and since x^* is a limit point, then

$$R(j(k)) \to -4\phi(x^*) + 0, \tag{36}$$

for $k \to \infty$. From (36), and Steps 4 and 5 of the APA, we conclude that any interval $[x_{j-1}, x_j]$, $j = j(k)$, whose characteristic satisfies condition (35) contains at least one point of the sequence $\{w^k\}$. On the contrary, any interval $[x_{i-1}, x_i]$, such that

$$R(i) < -4\phi(x^*) \tag{37}$$

does not contain any point of the sequence $\{w^k\}$.

By (26) we have for $m = k = K$

$$(w_0, w_1) = (x_0, x_1), \ldots, (w_{K-1}, w_K) = (x_{K-1}, x_K),$$

and so by (33 and 34), and decision rule (18), based on the series (14), we obtain that $\{w^k\} \subset \{x^m\}$. This inclusion makes it possible to estimate redundancy in terms of $\tau(m, n)$. Each redundant trial from $\tau(m, n)$, corresponds to an interval (x_{i-1}, x_i) satisfying inequality (37).

For the nonredundancy proof, $\tau(m, n) = 0$, we must show that at every iteration $m > n$, at least one interval exists, which characteristic satisfies (35) and the processor that completed its trial can take this interval to execute trial x^{m+p}.

After executing trial y^n, at every iteration, the intervals $[x_{j-1}, x_j]$ from (22) and $[x_{i-1}, x_i]$ from (29) exist. Of course, when one processor has completed its trial and chooses a new trial point, the second processor can continue to execute its current trial x^{k+p-1} within one of the intervals (22 and 29). Let us estimate their characteristics.

The characteristic of the interval $[x_{j-1}, x_j]$ satisfies the condition (35) because it contains the global minimizer x^* and (28) takes place. Let us estimate the characteristic $R(x_{i-1}, x_i)$ of the interval $[x_{i-1}, x_i]$.

From the Lipschitz condition, (27 and 29) we have

$$z_{i-1} + z_i \le \phi(x') + K_i(x' - x_{i-1}) + \phi(x') + K_i(x_i - x')$$

$$\le 2\phi(x^*) + 2\delta + K_i(x - x_{i-1}).$$

By substituting the obtained estimate in the formula (16) for the interval $[x_{i-1}, x_i]$, we deduce

$$R(x_{i-1}, x_i) \geq (x_i - x_{i-1})(r\mu + \mu_i^2(r\mu)^{-1} - 2K_i) - 4\delta - 4\phi(x^*),$$

where μ_i is from (30). From this inequality, the estimate (35) necessary follows if

$$\begin{cases} r\mu + \mu_i^2(r\mu)^{-1} - 2K_i > 0 \\ (x_i - x_{i-1})(r\mu + \mu_i^2(r\mu)^{-1} - 2K_i) > 4\delta \end{cases}$$

Solutions of the system are $r\mu$ and $x_i - x_{i-1}$ such that

$$r\mu > K_i + \sqrt{K_i^2 - \mu_i^2},$$

$$x_i - x_{i-1} > \tilde{\varepsilon},$$

where $\tilde{\varepsilon}$ is from (32). Therefore, for $r\mu$ satisfying (28), the condition (35) holds for the characteristic of the interval $[x_{i-1}, x_i]$ until its length is major than $\tilde{\varepsilon}$, which occurs while

$$x_t - x_{t-1} > \tilde{\varepsilon},$$

where t is from (17 and 18).

Thus, under assumptions of the theorem, when a processor having completed its previous iteration chooses a new trial point, at least one interval of the type (22) or (29) with a characteristic satisfying condition (35) exists. This means that no redundant trials will be produced by the APA, i.e., $\tau(m, n) = 0$. □

Corollary 3. If, under the conditions of the theorem, $n = 1$ and $\varepsilon > \tilde{\varepsilon}$, where ε and $\tilde{\varepsilon}$ are taken from (20) and (32), respectively, then the APA with two parallel processors executes the search of the global minimum up to the fulfillment of the stopping rule (20) without redundant trials.

Proof. The proof is evident and is so omitted. □

Corollary 4. Suppose that x^* and x^{**} ($x^* < x^{**}$) are the global minimizers of the function $\phi(x)$ and there exists a point $x^n \in \{x^k\}$ such that $x^* < x^n < x^{**}$ and relation (19) holds. Then the execution of the APA with $p = 2$ gives $\tau(m, n) = 0$ for any $m > n$.

Proof. The corollary is proved immediately by taking $\delta = 0$ in condition (27). □

Theorem 3 and its corollaries establish conditions for nonredundant parallelization of the APA. Note, that the theorem and corollaries are also true in the case when the computation times of function values differ significantly at different points of the search region. While one processor executes one trial, another processor can execute several trials during the same time without producing any redundancy.

5. COMPUTATIONAL RESULTS

In order to illustrate the theoretical considerations presented in the previous sections, we report the results of several numerical experiments connected with optimization of multidimensional multiextremal functions. We compare the purely sequential method from Strongin [34], the parallel synchronous algorithm from Sergeyev and Strongin [33], and the APA from Section 3. All the three methods are "nested" into scheme (4) for solving the multidimensional problems (2 and 3) and will be further called MSEQ, MSA, and MAA, respectively.

For implementing these experiments, we have worked out a program system simulating the execution of parallel MAA and MSA on a PC IBM computer with a single processor.

Three series of experiments were executed. Computational results are presented in three tables where the columns correspond to different combinations of the parallelization degrees (10) and the rows contain the relevant coefficients.

In the first two series of experiments, we introduce the following assumptions.

1. The time of the computation for the values $\phi(y)$ is the same for all feasible points $y \in D$. This time can be used as a "unit" or a "quantum" of the time during the search. Note, that this assumption is kind of the worst case for MAA, because it fixes the most advantageous situation for the synchronous method.

2. The computation time of the value $\phi(y)$ is much greater than the time that is spent for functioning the algorithm (i.e., the time spent to choose the new trial point and to communicate to the common search data base, the result of the trial).

Denote as

$$T_a(q) = T_a(q_1, q_2, \ldots, q_N), \qquad q = (q_1, q_2, \ldots, q_N),$$

the time (measured in computational quanta), which is required for solving by the MAA, problems (2 and 3) with q_i processors on the ith coordinate and as

$$T_s(q) = T_s(q_1, q_2, \ldots, q_N),$$

the similar time for the MSA.

It is obvious that for the purely sequential method MSEQ, the time of solving the problem is

$$T_1 = T_a(1, 1, \ldots, 1) = T_s(1, 1, \ldots, 1)$$

and it is equal just to the number of the function values (trials) computed. We denote as K_a the number of trials made by the asynchronous method MAA and as K_s the number of trials made by MSA.

To evaluate the speed up in time of the parallel methods in comparison with MSEQ, we introduce coefficients

$$u_a(q) = \frac{T_1}{T_a(q)}, \qquad u_s(q) = \frac{T_1}{T_s(q)},$$

where $u_a(q)$ measures the speed up obtained by the MAA and $u_s(q)$ by the MSA.

As a measure for a relative comparison of the efficiency of synchronism and asynchronism, we introduce the value

$$u(q) = \frac{u_a(q)}{u_s(q)} = \frac{T_s(q)}{T_a(q)}.$$

Note, that due to assumption 1, in the case $q = (1, 1, \ldots, 1, q_N)$ we have $u_a(q) = u_s(q)$ and $u(q) = 1$.

In the first series we maximized 100 two-dimensional functions

$$f(y) = \left\{ \left(\sum_{i=1}^{7} \sum_{j=1}^{7} (a_{ij} V_{ij}(y) + b_{ij} W_{ij}(y) \right)^2 + \left(\sum_{i=1}^{7} \sum_{j=1}^{7} (c_{ij} V_{ij}(y) - d_{ij} W_{ij}(y) \right)^2 \right\}^{1/2}, \tag{38}$$

where $y = (y_1, y_2) \in R^2$, $0 \le y_v \le 1$, $v = 1, 2$,

$$V_{ij}(y) = \sin(i\pi y_1)\ \sin(j\pi y_2),$$

$$W_{ij}(y) = \cos(i\pi y_1)\ \cos(j\pi y_2),$$

and a_{ij}, b_{ij}, c_{ij}, d_{ij} are random coefficients uniformly distributed on the interval $[-1, 1]$.

All the experiments have been performed with the accuracy $\varepsilon = 0.01$ from (20) and the parameter $r = 2$ for solving all the univariate subproblems (9). Moreover, all univariate searches started at the same initial points 0.2, 0.3, 0.7, 0.8.

For the purely sequential method MSEQ, the average number of trials obtained after maximizing 100 functions of type (38) (which, in this case, can be considered as a measure of the average time of the search) was $T_1 = 415.08$. The results obtained after applying the parallel methods are shown in Table I.

Table I. Average Results for 100 Two-Dimensional Functions of the Type (38)

q	(2, 1)	(2, 2)	(2, 3)	(2, 4)	(2, 5)
K_a	409.46	397.96	419.40	456.00	510.60
K_s	413.89	409.24	424.08	458.60	514.20
T_a	204.73	99.49	69.90	57.00	51.06
T_s	234.52	114.63	77.79	62.46	56.89
u_a	2.03	4.17	5.94	7.28	8.13
u_s	1.77	3.62	5.34	6.64	7.30
u	1.15	1.15	1.11	1.10	1.23
q	(3, 1)	(3, 2)	(3, 3)	(3, 4)	(3, 5)
K_a	427.23	429.18	432.90	472.20	529.95
K_s	432.40	426.16	433.83	469.44	527.90
T_a	123.41	71.53	48.10	39.35	35.33
T_s	174.38	84.90	55.95	44.63	40.62
u_a	2.91	5.80	8.63	10.55	11.75
u_s	2.38	4.89	7.42	9.30	10.22
u	1.22	1.19	1.16	1.13	1.15
q	(4, 1)	(4, 2)	(4, 3)	(4, 4)	(4, 5)
K_a	459.88	447.12	469.32	494.40	565.20
K_s	458.09	459.38	466.44	502.08	570.70
T_a	114.97	55.89	39.11	30.90	28.26
T_s	144.45	71.38	46.58	36.92	33.69
u_a	3.61	7.43	10.61	13.43	14.69
u_s	2.87	5.82	8.91	11.24	12.32
u	1.26	1.28	1.19	1.19	1.19

The second series of numerical experiments has been executed to minimize the test function from Lucidi and Piccioni [18]

$$f(y) = (\pi/N)\left\{10\ \sin^2(\pi y_1) + (y_N - 1)^2 + \sum_{i=1}^{N-1} [(y_j - 1)^2(1 + 10\ \sin^2(\pi y_{i+1}))]\right\} \tag{39}$$

The function is defined over the region

$$D = \left\{y \in R^N :\ -2 \le y_i \le 4, 1 \le i \le N\right\} \tag{40}$$

for different dimensions N. As the parameters of the algorithms, we used in all cases the accuracy $\varepsilon = 0.12$, the reliability parameter $r = 2$, and initial stage of the search in all the univariate subproblems (9) consisted in the execution of four first trials at points $\{-1.2, -0.5, 0.0, 2.2\}$. Computational results of the compared methods are described in Table II.

In the last series of experiments, we omit assumption 1 and study the situation when the time of evaluation of the objective function depends on the trial point. We used in all cases the function (39) with $N = 3$ defined over the region (40), the accuracy $\varepsilon = 0.01$, the reliability parameter $r = 2$, and initial stage of the search in all the univariate subproblems (9) consisted in the execution of four first trials at points $\{-1.2, -0.5, 0.0, 2.2\}$. Computational results of the compared methods are described in Table III. We consider two cases:

Case A. The time of evaluation of $f(y)$ at a point y is equal to

$$t_1(f(y)) = \text{int}(999(f(y) - f_{\min})/(f^{\max} - f_{\min})),$$

where $\text{int}(x)$ is the integer part of x and $f_{\min} = 0, f^{\max} = 1.85$.

Case B. The time of evaluation of $f(y)$ at a point y is equal to

$$t_2(f(y)) = 1000 - t_1(f(y)).$$

It can be seen from Table III that in both cases only a small number of redundant trials appears and the asynchronous method works faster than the synchronous one.

Table II. Numerical Results for the Multidimensional Functions (39)

	q	(1, 1, 1)	(2, 2, 2)	(3, 3, 3)	(4, 4, 4)
$N = 3$	K_a	7791	7056	7884	10368
	K_s	7791	6772	7548	10836
	T_a	7791	882	292	162
	T_s	7791	1013	390	216
	u_a	1.00	8.83	26.68	48.09
	u_s	1.00	7.69	19.98	36.07
	u	1.00	1.15	1.34	1.33
	q	(1, 1, 1, 1)	(2, 2, 2, 2)	(3, 3, 3, 3)	(4, 4, 4, 4)
$N = 4$	K_a	155694	136576	165807	201216
	K_s	155694	130168	152895	232682
	T_a	155694	8536	2047	786
	T_s	155694	10658	3062	1296
	u_a	1.00	18.24	76.06	198.08
	u_s	1.00	14.61	50.85	120.13
	u	1.00	1.25	1.50	1.65
	q	(1, 1, 1, 1, 1)	(2, 2, 2, 2, 2)	(3, 3, 3, 3, 3)	(4, 4, 4, 4, 4)
$N = 5$	K_a	3111771	2635872	3258144	4786176
	K_s	3111771	2502128	3097143	4996416
	T_a	3111771	82371	13408	4674
	T_s	3111771	112485	24119	7776
	u_a	1.00	37.78	232.08	655.76
	u_s	1.00	27.66	129.02	400.18
	u	1.00	1.36	1.80	1.67

Table III. Numerical Results for the Multidimensional Functions (39) When the Time of Evaluating $f(y)$ Is Not the Same Over the Search Region

	Case A		Case B	
q	(2, 2, 2)	(3, 3, 3)	(2, 2, 2)	(3, 3, 3)
K_a	106802	117270	105599	111774
K_s	101510	103671	101510	103671
T_a	738361	221739	11900017	3629895
T_s	1527229	791559	16943876	5769930
u	2.07	3.57	1.42	1.59

6. CONCLUDING REMARKS

In this paper the parallel asynchronous information algorithm for solving multidimensional Lipschitz global optimization problems has been proposed. The new method uses the nested optimization scheme and the new parallel asynchronous global optimization method for solving the core univariate subproblems generated during the search by the nested scheme.

It has been shown that during use of this scheme, the computations became essentially asynchronous, even in the case when the original multidimensional problem had equal times of the objective function evaluations over the whole search region. The application of asynchronous global optimization methods becomes crucial when these times are different from point to point.

Global convergence conditions for the new method have been established. Theoretical conditions of speed up, which can be reached by using the new asynchronous global optimization method in comparison with the pure sequential case, have been presented.

Three series of numerical experiments comparing sequential, synchronous, and asynchronous algorithms were also reported. The obtained results show that both asynchronous and synchronous schemes work significantly better on the considered test functions than the sequential one. Of course, this result takes place until the redundant points appear. Since the functions considered in global optimization problems usually have many local optima, Theorem 3 ensures good levels of speed up for the new method.

The numerical experiments confirm the theoretical considerations regarding the nested optimization scheme and show that asynchronous algorithm works better in comparison with the synchronous technique for all the parallelization degrees used. In all experiments, it provides the speed up $u_a > u_s$ and this advantage attains 80% (the second series of experiments, case $N = 5$, $q_i = 3$, $1 \leq i \leq 5$; see Table II). Moreover, the results demonstrate the trend of the increasing of efficiency of the asynchronous algorithm when the dimension and/or the number of processors assigned to one coordinate increases.

ACKNOWLEDGMENT

The authors thank the anonymous referees for their useful suggestions.

REFERENCES

1. F. Aluffi-Pentini, V. Parisi, and F. Zirilli, Global optimization and stochastic differential equations, *J. Optimization Theory Appl.* 47, 1–16 (1985).

2. F. Archetti and F. Schoen, A survey on the global optimization problems: general theory and computational approaches, *Ann. Operations Res.* 1, 87–110 (1984).
3. W. Baritompa, Accelerations for a variety of global optimization methods, *J. Global Optimization* 4, 37–45 (1994).
4. M. Bertocchi, A parallel algorithm for global optimization, *Optimization* 62, 379–386 (1990).
5. D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, Englewood Cliffs, 1989.
6. I. M. Bomze, T. Csendes, R. Horst, and P. M. Pardalos, eds., *Developments in Global Optimization*, Kluwer Academic Publ., Dordrecht, 1997.
7. C. R. Carr and C. W. Howe, *Quantitative Decision Procedures in Management and Economics. Deterministic Theory and Applications*, McGraw Hill, New York, 1964.
8. L. C. W. Dixon and G. P. Szego, eds., *Towards Global Optimization*, Vol. 2. North-Holland, Amsterdam, 1978.
9. C. A. Floudas and P. M. Pardalos, eds., *State of the Art in Global Optimization*, Kluwer Academic Publ., Dordrecht, 1996.
10. V. P. Gergel and Ya. D. Sergeyev, Sequential and parallel global optimization algorithms using derivatives, *Computers Math. Appl.*, 37, 163–189 (1999).
11. V. A. Grishagin, On convergence conditions for a class of global search algorithms, *Proc. 3rd All-Union Seminar Numerical Methods Nonlinear Programming*, Kharkov, 82–84 (1979) (in Russ.).
12. V. A. Grishagin, Ya. D. Sergeyev, and R. G. Strongin, Parallel characteristic algorithms for solving problems of global optimization, *J. Global Optimization* 10, 185–206 (1997).
13. P. Hansen and B. Jaumard, Lipschitz optimization, in *Handbook of Global Optimization*, R. Horst and P. M. Pardalos, eds., Kluwer Academic Publ., Dordrecht, 1995.
14. E. M. T. Hendrix, Ph.D. thesis, Wageningen, 1998.
15. R. Horst and P. M. Pardalos, eds., *Handbook of Global Optimization*, Kluwer Academic Publ., Dordrecht, 1995.
16. R. Horst and H. Tuy, *Global Optimization Deterministic Approaches*, 3rd ed., Springer Verlag, Berlin, 1996.
17. R. B. Kearfott, *Rigorous Global Search: Continuous Problems*, Kluwer Academic Publ., Dordrecht, 1996.
18. S. Lucidi and M. Piccioni, Random tunneling by means of acceptance–rejection sampling for global optimization, *J. Optimization Theory Appl.* 62, 255–277 (1989).
19. A. Migdalas, P. M. Pardalos, and S. Støroy, eds., *Parallel Computing in Optimization*, Kluwer Academic Publ., Dordrecht, 1996.
20. R. H. Mladineo, An algorithm for finding the global maximum of a multimodal, multivariate function, *Math. Programming* 34, 188–200 (1986).
21. P. M. Pardalos, Parallel search algorithms in global optimization, *Applied Math. Computation* 29, 219–229 (1989).
22. P. M. Pardalos, A. T. Phillips, and J. B. Rosen, *Topics in Parallel Computing in Mathematical Programming*, Science Press, 1992.
23. P. M. Pardalos and J. B. Rosen, eds., Computational methods in global optimization, *Ann. Operations Res.* 25 (1990).
24. S. A. Pijavskii, An algorithm for finding the absolute extremum of a function, *USSR Comput. Maths Math. Phys.* 12, 57–67 (1972).
25. J. Pintér, Convergence qualification of adaptive partition algorithms in global optimization, *Math. Programming* 56, 343–360 (1992).
26. J. Pintér, *Global Optimization in Action*, Kluwer Academic Publ., Dordrecht, 1996.
27. H. Ratschek and J. Rokne, *New Computer Methods for Global Optimization*, Ellis

Horwood, Chichester, 1988.
28. A. H. G. Rinnooy Kan and G. H. Timmer, Global optimization, *Optimization* (G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, eds.), North-Holland, Amsterdam, 1989.
29. Ya. D. Sergeyev, An information global optimization algorithm with local tuning, *SIAM J. Optimization* 5, 615–631 (1995).
30. Ya. D. Sergeyev, An information global optimization algorithm with local tuning, *J. Global Optimization* 15, 157–167 (1999).
31. Ya. D. Sergeyev and V. A. Grishagin, A parallel algorithm for finding the global minimum of univariate functions, *J. Optimization Theory Appl.* 80, 513–536 (1994a).
32. Ya. D. Sergeyev and V. A. Grishagin, Sequential and parallel algorithms for global optimization, *Optimization Methods Software* 3, 111–124 (1994b).
33. Ya. D. Sergeyev and R. G. Strongin, A global minimization algorithm with parallel iterations, *USSR Comput. Maths. Math. Phys.* 29, 7–15 (1990).
34. R. G. Strongin, *Numerical Methods on Multiextremal Problems*, Nauka, Moscow, 1978 (in Russ.).
35. R. G. Strongin, Algorithms for multi-extremal mathematical programming problems employing the set of joint space-filling curves, *J. Global Optimization* 2, 357–378 (1992).
36. R. G. Strongin and Ya. D. Sergeyev, Global multidimensional optimization on parallel computer, *Parallel Comput.* 18, 1259–1273 (1992).
37. A Törn and A. Žilinskas, *Global Optimization,* Lecture Notes in Computer Science, Springer Verlag, New York, 1989, p. 350.
38. G. R. Wood, The bisection method in higher dimensions, *Math. Programming* 55, 319–337 (1992).
39. Z. B. Zabinsky and R. L. Smith, Pure adaptive search in global optimization, *Math. Programming* 53, 323–338 (1992).
40. A. A. Zhigljavsky, *Theory of Global Random Search*, Kluwer Academic Publ., Dordrecht, 1991.