VYTAUTAS MAGNUS UNIVERSITY
INSTITUTE OF MATHEMATICS AND INFORMATICS

Remigijus PAULAVIČIUS

GLOBAL OPTIMIZATION WITH SIMPLICIAL PARTITIONS

Summary of Doctoral Dissertation

Physical Sciences (P 000)
Informatics (09 P)
Informatics, Systems Theory (P 175)

Vilnius, 2010

Doctoral dissertation was prepared at the Institute of Mathematics and Informatics in 2006–2010.

Scientific Supervisor

Dr (HP) Julius ŽILINSKAS (Institute of Mathematics and Informatics, Physical Sciences, Informatics – 09P).

The dissertation is being defended at the Council of Scientific Field of Informatics at Vytautas Magnus University:

Chairman

Prof Dr Habil Vytautas KAMINSKAS (Vytautas Magnus University, Physical Sciences, Informatics – 09 P).

Members:

Prof Dr Habil Gintautas DZEMYDA (Institute of Mathematics and Informatics, Physical Sciences, Informatics – 09 P),

Prof Dr Habil Artūras KAKLAUSKAS (Vilnius Gediminas Technical University, Physical Sciences, Informatics – 09 P),

Prof Dr Habil Jonas MOCKUS (Institute of Mathematics and Informatics, Technological Sciences, Informatics Engineering – 07 T),

Prof Dr Habil Rimantas ŠEINAUSKAS (Kaunas University of Technology, Technological Sciences, Informatics Engineering – 07 T).

Opponents:

Prof Dr Romas BARONAS (Vilnius University, Physical Sciences, Informatics – 09 P),

Prof Dr Habil Leonidas SAKALAUSKAS (Institute of Mathematics and Informatics, Physical Sciences, Informatics – 09 P).

The dissertation will be defended at the public meeting of the Council of Scientific Field of Informatics in the auditorium number 203 of the Institute of Mathematics and Informatics at 1 p. m. on September 29 2010.
Address: Akademijos str. 4, LT-08663 Vilnius, Lithuania.
Tel. +370 5 210 9300; fax +370 5 270 0112;
e-mail: mathematica@ktl.mii.lt

The summary of the dissertation was distributed on August 27 2010.
A copy of the doctoral dissertation is available for review at the Martynas Mažvydas National Library of Lithuania, the Library of the Vytautas Magnus University and at the Library of Institute of Mathematics and Informatics.

VYTAUTO DIDŽIOJO UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS INSTITUTAS

Remigijus PAULAVIČIUS

GLOBALUSIS OPTIMIZAVIMAS SU SIMPLEKSINIAIS POSRIČIAIS

Daktaro disertacijos santrauka

Fiziniai mokslai (P 000)
Informatika (09 P)
Informatika, sistemų teorija (P 175)

Vilnius, 2010

Disertacija rengta 2006–2010 metais Matematikos ir informatikos institute.

Mokslinis vadovas

dr. (HP) Julius ŽILINSKAS (Matematikos ir informatikos institutas, fiziniai mokslai, informatika – 09 P).

Disertacija ginama Vytauto Didžiojo universiteto Informatikos mokslo krypties taryboje:

Pirmininkas

prof. habil. dr. Vytautas KAMINSKAS (Vytauto Didžiojo universitetas, fiziniai mokslai, informatika – 09 P).

Nariai:

prof. habil. dr. Gintautas DZEMYDA (Matematikos ir informatikos institutas, fiziniai mokslai, informatika – 09 P),

prof. habil. dr. Artūras KAKLAUSKAS (Vilniaus Gedimino technikos universitetas, fiziniai mokslai, informatika – 09 P),

prof. habil. dr. Jonas MOCKUS (Matematikos ir informatikos institutas, technologijos mokslai, informatikos inžinerija – 07 T),

prof. habil. dr. Rimantas ŠEINAUSKAS (Kauno technologijos universitetas, technologijos mokslai, informatikos inžinerija – 07 T).

Oponentai:

prof. dr. Romas BARONAS (Vilniaus universitetas, fiziniai mokslai, informatika – 09 P),

prof. habil. dr. Leonidas SAKALAUSKAS (Matematikos ir informatikos institutas, fiziniai mokslai, informatika – 09 P).

Disertacija bus ginama viešame Informatikos mokslo krypties tarybos posėdyje 2010 m. rugsėjo 29 d. 13 val. Matematikos ir informatikos institute, 203 auditorijoje.

Adresas: Akademijos g. 4, LT-08663 Vilnius, Lietuva.

Tel.: (8 5) 21 09 300 faksas (8 5) 27 29 209;

el. paštas: mathematica@ktl.mii.lt

Disertacijos santrauka išsiuntinėta 2010 m. rugpjūčio 27 d.

Disertaciją galima peržiūrėti Martyno Mažvydo nacionalinėje bibliotekoje, Vytauto Didžiojo universiteto ir Matematikos ir informatikos instituto bibliotekose.

Introduction

Research area

The methods of global optimization are used for problem solving in such fields as computational chemistry and biology, biomedicine, operation research, economics, engineering design and management as well as numerous other engineering and applied sciences. One of the most researched fields of global optimization is Lipschitz optimization which is based on the assumption that the objective function satisfies the Lipschitz condition. The multidimensional methods of Lipschitz optimization are less researched than the one-dimensional methods both theoretically and practically. These methods usually use rectangular partitions and simply calculated Lipschitz bounds. Less attention is paid to the simplicial partitions and the development of bounds. Therefore, the research area of this work is the multidimensional simplicial algorithms of Lipschitz optimization that use aggregate Lipschitz bounds with various norms.

Topicality of the work

The rapidly developing information technologies and the progress of computer technology allow solving the steadily increasing number of complex optimization problems, the solvability of which has been nearly impossible until recently. The global optimization methods based on the calculation of the Lipschitz bounds have been analyzed in thorough detail and are widely applied in solving various optimization problems. The application of these methods is determined by the fact that they provide the same results every time, which is different in case of stochastic optimization methods. The Lipschitz methods, contrary to the direct search and stochastic methods, guarantee the precision of the found solution during the finite (may as well be long) period of time. Moreover, differently from the interval methods, these methods can be used in the "black box" situation.

However, if compared to other global optimization methods, the Lipschitz methods have some important disadvantages as well. The Lipschitz methods are usually based on the assumption that the Lipschitz constant is known ahead of time, which is a rare case in solving practical problems. Furthermore, the Lipschitz algorithms are slower than some of the other global optimization algorithms. When the dimension number n increases, the calculation scope of Lipschitz algorithm increases exponentially.

Therefore, the main part of the present dissertation work is devoted to the creation of improved bounds that determine faster Lipschitz algorithms. Also, an important part of the dissertation focuses on the implementation and evaluation of the proposed effective parallel versions of the algorithms, executed with the purpose of solving larger optimization problems.

Research object

The research object of the work is as follows:

- Lipschitz global optimization multidimensional algorithms with simplicial partitions;
- Application of parallel computer systems in global optimization.

The aim and tasks of the dissertation

The aim of the thesis is to modify the existing algorithms of the Lipschitz global optimization in the multidimensional case and to propose new ones in order to solve optimization problems faster and more precisely, also using the parallel computer systems to solve more complex optimization problems. The following tasks were formulated in order to reach the aim:

1. To review the existing global optimization algorithms and to define the algorithm group which is to be researched;
2. To investigate how the results of optimization are influenced by various norms and their corresponding Lipschitz constants used to form the Lipschitz bounds.
3. To use simplicial partitions in order to create:
   - Corrected trivial Lipschitz bounds which use various norms and their corresponding Lipschitz constants.
   - The Piyavskii type Lipschitz bounds which are obtained when solving linear equation systems in multidimensional cases.
   - The Lipschitz bounds based on the radius of the circumscribed multidimensional sphere.
   - Aggregate Lipschitz bounds composed of a few different Lipschitz bound types with various norms.
4. To create sequential and parallel simplicial branch and bound algorithms of Lipschitz optimization with proposed bounds.

5. To compare the efficiency of the created algorithms using various proposed Lipschitz bounds.
6. To compare the achieved results with the other well known Lipschitz optimization algorithms.
7. To investigate the efficiency of the created sequential and parallel algorithms using various selection strategies: best first, breadth first, depth first and statistical.
8. To compare the efficiency of the created parallel algorithms intended for the shared and distributed memory computers.

Scientific novelty

According to the results of the theoretical and experimental investigations, the Lipschitz optimization results are influenced by the used norms and their corresponding Lipschitz constants. On the basis of the obtained results, it was proposed to use several norms instead of one particular norm and combine the bounds of different types with various norms when calculating the trivial Lipschitz bounds. This way, more precise Lipschitz bounds are obtained without calculating the objective function in additional points.

When using the first norm, it was proposed how to calculate the Piyavskii type bounds in solving the linear equation systems instead of quadratic equations in case of the Euclidean norm.

In the case of the Euclidean norm, a new Lipschitz bound based on the radius of a circumscribed multidimensional sphere was proposed.

The efficiency of the proposed simplicial branch and bound algorithm was tested using various proposed Lipschitz bounds and then compared to the other well known Lipschitz algorithms.

The algorithm efficiency was investigated depending on the used selection strategy (best first, breadth first, depth first and statistical) according to the criteria of number of function evaluations, optimization time, total number of elements and maximal size of candidate list.

Parallel implementations of the proposed sequential algorithm were created for the shared memory (using the OpenMP standard) and the distributed memory (using the MPI standard) computers.

The software was developed based on the created algorithms:

- Sequential version:
  http://www.gridglobopt.vgtu.lt/files/AT_Projekto_

Ataskaita/BranchBoundTemplate/BBSequential/Examples/SimplicialBBSequential/

- Parallel version intended for the shared memory computers: http://www.gridglobopt.vgtu.lt/files/AT_Projekto_Ataskaita/BranchBoundTemplate/BBParallel/Examples/SimplicialBBOpenMP/
- Parallel version intended for the distributed memory computers: http://www.gridglobopt.vgtu.lt/files/AT_Projekto_Ataskaita/BranchBoundTemplate/BBParallel/Examples/SimplicialBBParallel/

Defended propositions

1. More precise bounds are obtained when using the aggregate Lipschitz bounds composed of different bounds with various norms; due to this, the optimization algorithms are accelerated.
2. Calculation of the Piyavskii type bounds using the first norm enables the replacement of the quadratic equation system solving into the linear equation system solving.
3. The Lipschitz bounds which are based on the radius of a circumscribed multidimensional sphere are most frequently better than the ones usually used in the branch and bound algorithms.
4. There is no one particular selection strategy which could be used to obtain the best optimization results according to all of the researched criteria.
5. After vertex verification, the number of function evaluations decreases considerably and the efficiency of the parallel algorithm intended for the shared memory computers increases.
6. The number of function evaluations of the parallel algorithm intended for the distributed memory computers with the best found function value interchange or without it differ only slightly.

The scope of the scientific work

The present dissertation consists of an introduction, three main parts and the general conclusions. The appendixes of the paper include a list of used notations and abbreviations, the optimization problems used in the

experimental researches, glossary, index and the list of references. The scope of the paper is 134 pages including 20 pictures, 15 tables and 5 algorithms. The list of references consists of 135 literature sources.

1. Global Optimization Methods

In this chapter, we give a description of the dissertation topics and review of the main results, obtained by other researchers.

Many problems in engineering, physics, economics and other fields may be formulated as optimization problems, where the maximum value of an objective function must be found. The standard global optimization problem is to find f^* such that

$$f^* = \max_{x \in \mathbf{D}} f(x),$$

where the objective function $f(x), f : \mathbb{R}^n \to \mathbb{R}$, is a nonlinear function of continuous variables, $\mathbf{D} \subset \mathbb{R}^n$ is a feasible region, n is the number of variables. Besides the global optimum f^* one or all global optimizers $x^* : f(x^*) = f^*$ must be found or shown that such a point does not exist. In our cases, $\mathbf{D}$ is compact and f is a Lipschitz function, therefore the existence of x^* assured by the well-known theorem of Weierstrass. In Lipschitz optimization only a point $x_{opt} \in \mathbf{D}$ such that $f(x_{opt})$ differs from f^* by no more than a specified accuracy ε can be found.

Branch and bound method

Branch and bound is a technique for the implementation of covering global optimization methods as well as combinatorial optimization algorithms. An iteration of a classical branch and bound algorithm processes a node in the search tree representing a not yet explored subspace of the solution space. The iteration has three main components: selection of the node to process, branching of the search tree and bound calculation. Branching is implemented by division of the subspaces. The algorithm detects subspaces, which cannot contain a global optimizer, by evaluating bounds for the optimum over considered subspaces. Subspaces which cannot contain a global maximum are discarded from further search pruning the branches of the search tree. The rules of selection, branching and bounding differ from algorithm to algorithm.

Lipschitz optimization algorithms

Lipschitz optimization is one of the most deeply investigated subjects of global optimization. A function $f : \mathbf{D} \to \mathbb{R},\ \mathbf{D} \subset \mathbb{R}^n$, is said to be Lipschitz if it satisfies the condition

$$|f(x_1) - f(x_2)| \leq L \|x_1 - x_2\|,\ \forall x_1, x_2 \in \mathbf{D}, \tag{1}$$

where $L > 0$ is a Lipschitz constant, $\mathbf{D}$ is compact and $\|\cdot\|$ denotes a norm.

Convergent deterministic Lipschitz optimization methods fall into three main classes. First, multivariate Lipschitz optimization can be reduced to the univariate case. Following this idea, a nested optimization scheme and filling the feasible region by Peano curve were proposed.

The second class contains direct extensions of Piyavskii's method to the multivariate case. Various modifications using the Euclidean or other norms or close approximations have been proposed. Most of these algorithms can be improved by interpreting them using branch and bound method.

The third class contains many branch and bound algorithms, but, in general, considerably weaker bounds. These algorithms fit into the general framework proposed by Horst (1986). The algorithms differ in the selection rules, the ways subdivision is performed and bounds are computed.

In general, Lipschitz bounds belong to the following three families: φ, μ_1 and μ_2. The sharpest upper bound over sub-region $\mathbf{I} \subseteq \mathbf{D}$, given the function values $f(x_i), x_i \in \mathbf{T}$, and the Lipschitz constant L, is provided by

$$\varphi(\mathbf{I}) = \max_{x \in \mathbf{I}} \min_{x_i \in \mathbf{T}} \{f(x_i) + L\|x - x_i\|\}. \tag{2}$$

Many univariate algorithms use the bound φ, where the set $\mathbf{T}$ is suitably updated in an iterative way. The most studied of these methods is due to Piyavskii. For $(n > 2)$, however, problem (2) constitutes a difficult optimization problem.

The branch and bound algorithms use considerably weaker bounds. Let

$$\delta(\mathbf{I}) = \max\{\|x_1 - x_2\| : x_1, x_2 \in \mathbf{I}\}$$

denote the diameter of $\mathbf{I} \subset \mathbf{D}$. For example, if $\mathbf{I} = \{x \in \mathbb{R}^n : a \leq x \leq b\}$ is an n-rectangle, then $\delta(\mathbf{I}) = \|b - a\|$, and if $\mathbf{I}$ is an n-simplex, then the diameter $\delta(\mathbf{I})$ is the length of its longest edge. Afterwards a simple upper bound can be derived from (1):

$$\mu_1(\mathbf{I}) = \min_{x_i \in \mathbf{T}} f(x_i) + L\delta(\mathbf{I}), \tag{3}$$

where $\mathbf{T} \subset \mathbf{I}$ is a finite sample of points in $\mathbf{I}$, where the function values of f have been evaluated. If $\mathbf{I}$ is a rectangle or a simplex, the set $\mathbf{T}$ often coincides with the vertex set $V(\mathbf{I})$. A tighter but computationally more expensive bound than (3) is

$$\mu_2(\mathbf{I}) = \min_{x_i \in \mathbf{T}} \left\{ f(x_i) + L \max_{x \in \mathbf{I}} \|x - x_i\| \right\}. \tag{4}$$

2. Lipschitz Optimization with Simplicial Partitions

Performance of the branch and bound algorithms depends on tightness of bounds. Bounds may be estimated using convex envelopes, interval arithmetic as well as the Lipschitz condition. In this chapter, improved and new Lipschitz bounds with various norms over simplices are proposed.

Norms and their corresponding Lipschitz constants

The value of Lipschitz constant depends on the used norm. In Lipschitz optimization the Euclidean norm is used most often, but other norms can also be considered. In this section, we formulate a proposition in which we present the dependence between Lipschitz constant and used norm.

Proposition 1. *Let $\mathbf{D} \subset \mathbb{R}^n$ be convex bounded closed set, and let $f(x) : \mathbf{D} \to \mathbb{R}$ be continuously differentiable function on an open set containing $\mathbf{D}$. Then $f(x)$ is Lipschitz function and $\forall x_1, x_2 \in \mathbf{D}$ is a true inequality*

$$|f(x_1) - f(x_2)| \leq L_p \|x_1 - x_2\|_q,$$

where $L_p = \max\{\|\nabla f(x)\|_p : x \in \mathbf{D}\}$ is a Lipschitz constant and $\frac{1}{p} + \frac{1}{q} = 1$, $1 \leq p, q \leq \infty$.

Corrected μ_2 type Lipschitz bounds with various l_p norms

In this section, we investigate how the results of optimization are influenced by the used norms and their corresponding Lipschitz constants when μ_2 type bound (4) is used.

In one-dimensional case, all norms are equal. But in multidimensional case evaluated bounds depend on the used norm. Investigations have shown, that no single norm and their corresponding Lipschitz constant is the best for all optimization problems and better results may be achieved when non

Euclidean norms are used. Therefore, for better upper bound we propose to use several norms: first and infinite instead of one particular norm:

$$\mu_2^{1,\infty}(\mathbf{I}) = \min_{v_i \in V(\mathbf{I})} \left\{ f(v_i) + \min \left\{ L_\infty \max_{x \in V(\mathbf{I})} \|x - v_i\|_1, L_1 \max_{x \in V(\mathbf{I})} \|x - v_i\|_\infty \right\} \right\}.$$

The Piyavskii (φ) type Lipschitz bound with the first norm

In this section, Piyavskii type bound (2) based on the first norm (φ^1) was proposed. In this case, the upper bounding function is the envelope of n-dimensional pyramids and its maximum point is found by solving a system of linear equations. In the case of the Euclidean norm, the upper bounding function is the envelope of n-dimensional cones and its maximum point is found by solving a system of quadratic and linear equations. Therefore, the bound based on the first norm is less computationally expensive.

Let us formulate two propositions, which are used for evaluation of φ^1.

Proposition 2. *If two n-pyramids* $F_{v_1}(x) = f(v_1) + L_\infty \|x - v_1\|_1$ *and* $F_{v_2}(x) = f(v_2) + L_\infty \|x - v_2\|_1$ *are defined and* $f(v_1) \geq f(v_2)$ *then the intersection of pyramids is contained in manifold of dimensionality* $n-1$ *defined by*

$$\sum_{i=1}^{n} d(v_{1i}, v_{2i}) - \frac{f(v_2) - f(v_1)}{L_\infty} = 0, \tag{5}$$

where

$$d(v_{1i}, v_{2i}) = \begin{cases} |2x_i - v_{1i} - v_{2i}| & \text{when } v_{1i} \leq x_i \leq v_{2i} \\ 0 & \text{when } v_{1i} = v_{2i} \\ |v_{1i} - v_{2i}| & \text{when } x_i \notin [v_{1i}, v_{2i}] \end{cases}$$

and all points x in the intersection are closer to the vertex v_1 than to v_2, i.e.

$$||x - v_1||_1 \leq ||x - v_2||_1.$$

Proposition 3. *The Piyavskii type Lipschitz bound with the first norm φ^1 can be found solving a system of n linear equations.*

The Lipschitz bound based on the radius of the circumscribed multidimensional sphere

Calculation of tight Lipschitz bounds usually is computationally expensive. Therefore, it is important to investigate possibilities of bounds tighter than μ_1(3) and μ_2(4), but computationally less expensive than φ(2). In this section, we propose new bound based on radius of the circumscribed sphere and function values at vertices.

Proposition 4. *Let* $\mathbf{I} \subset \mathbb{R}^n$ *is an* n*-simplex,* $R(\mathbf{I})$ *denote the radius of the circumscribed sphere and* $V(\mathbf{I})$ *is the vertex set. Then*

$$\psi^2(\mathbf{I}) = \max_{v \in V(\mathbf{I})} f(v) + LR(\mathbf{I}),$$

overestimate $f(x), \forall x \in \mathbf{I}$.

Sequential and parallel branch and bound algorithms with simplicial partitions and various Lipschitz bounds

In this section, a sequential and parallel branch and bound algorithms with simplicial partition and various Lipschitz bounds were proposed.

For simplicial branch and bound, the feasible region should be initially covered by simplices. The most preferable initial covering is face to face vertex triangulation - partitioning of the feasible region into finitely many n-dimensional simplices, whose vertices are also the vertices of the feasible region. We use a standard way of triangulation into $n!$ simplices. All simplices share the diagonal of the feasible region and are of equal hypervolume. There are several ways to divide the simplex into subsimplices. Experiments have shown, that the most preferable partitioning is a subdivision of the simplex into two by a hyper-plane passing through the middle point of the longest edge and the vertices which do not belong to the longest edge. The lower and upper bounds for the maximum of the function over a simplex are estimated using function values at vertices. Proposed sequential branch and bound algorithm with simplicial partition and various Lipschitz bounds is shown in Algorithm 1.

The solution of multidimensional Lipschitz optimization problem requires a lot of computing time and memory resources. In this section, we create a parallel OpenMP and MPI versions of proposed sequential branch and bound algorithm.

Two OpenMP versions were implemented: with a new vertex point verification in the global vertex set of all previously evaluated function values at the midpoints of the longest edge and without it. If the function value at

Algorithm 1 Simplicial branch and bound with various Lipschitz bounds

1: Cover feasible region $\mathbf{D}$ by $\mathbb{I} \leftarrow \{\mathbf{I}_j | \mathbf{D} = \cup \mathbf{I}_j, j = 1, \ldots, n! \}$ using face-to-face vertex triangulation
2: $\mathbf{S} \leftarrow \emptyset$, $LB(\mathbf{D}) \leftarrow -\infty$
3: **while** $(\mathbb{I} \neq \emptyset)$ **do**
4: Choose $\mathbf{I}_j \in \mathbb{I}$ using selection strategy, $\mathbb{I} \leftarrow \mathbb{I} \backslash \{\mathbf{I}_j\}$
5: $LB(\mathbf{D}) \leftarrow \max\{LB(\mathbf{D}), \max\limits_{v \in V(\mathbf{I}_j)} f(v)\}$
6: $\mathbf{S} \leftarrow \arg\max\{f(\mathbf{S}), \max\limits_{v \in V(\mathbf{I}_j)} f(v)\}$
7: calculate $UB(\mathbf{I}_j)$ using different proposed Lipschitz bounds
8: **if** $(UB(\mathbf{I}_j) - LB(\mathbf{D}) > \varepsilon)$ **then**
9: Branch $\mathbf{I}_j$ into 2 simplices: $\mathbf{I}_{j_1}$, $\mathbf{I}_{j_2}$
10: $\mathbb{I} \leftarrow \mathbb{I} \cup \{\mathbf{I}_{j_1}, \mathbf{I}_{j_1}\}$
11: **end if**
12: **end while**

the new vertex point has not been evaluated before, then a function value is evaluated. In the other case, the previously evaluated function value is used to calculate bounds. Therefore, it is possible to reduce the number of function evaluations avoiding several evaluations at the same point.

The OpenMP version of the parallel branch and bound algorithm is shown in Algorithm 2. Data parallelism is used. The feasible region $\mathbf{D}$ is subsequently divided into a set of simplices $\mathbb{I} = \{\mathbf{I}_j\}$. In C/C++, OpenMP directives are specified by using the #pragma mechanism provided by the C and C++ standards. Directive "for" specifies that the iterations of the loop immediately following it must be executed in parallel by different threads. "schedule(static)" describes that iterations of the loop are evenly (if possible) divided and then statically assigned to threads. The directive "critical" specifies a region of code that must be executed by only one thread at a time.

Two MPI versions with static load balancing were implemented using a parallel branch and bound template developed at Vilnius Gediminas Technical University: with interchange of the best currently found values among processors and without it. When the template is used, only algorithm specific rules should be described by the user and the standard parts are implemented in the template.

Static load balancing is used: tasks are initially distributed evenly (if possible) among p processors. Each parallel processor runs the same algorithm, which is shown in Algorithm 3. The algorithm is very similar to the sequential algorithm. The differences are:

Algorithm 2 Parallel (OpenMP) branch and bound with various Lipschitz bounds

Cover feasible region $\mathbf{D}$ by $\mathbb{I} \leftarrow \{\mathbf{I}_j | \mathbf{D} = \cup \mathbf{I}_j, j = 1, \ldots, n! \}$ using face-to-face vertex triangulation
$\mathbf{S} \leftarrow \emptyset$, $LB(\mathbf{D}) \leftarrow -\infty$
while $(\mathbb{I} \neq \emptyset)$ **do**
  $k = |\mathbb{I}|$
  #pragma omp parallel private (UB)
  #pragma omp for schedule (static)
  for $(j = 0; j <= k; j{+}{+})$ **do**
    Choose $\mathbf{I}_j \in \mathbb{I}$ using selection strategy, $\mathbb{I} \leftarrow \mathbb{I} \backslash \{\mathbf{I}_j\}$
    #pragma omp critical $(LB(\mathbf{D}))$
    $LB(\mathbf{D}) \leftarrow \max\{LB(\mathbf{D}), \max\limits_{v \in V(\mathbf{I}_j)} f(v)\}$
    #pragma omp critical $(\mathbf{S})$
    $\mathbf{S} \leftarrow \arg\max\{f(\mathbf{S}), \max\limits_{v \in V(\mathbf{I}_j)} f(v)\}$
    calculate $UB(\mathbf{I}_j)$ using different proposed Lipschitz bounds
    if $(UB(\mathbf{I}_j) - LB(\mathbf{D}) > \varepsilon)$ **then**
      Branch $\mathbf{I}_j$ into 2 simplices: $\mathbf{I}_{j_1}$, $\mathbf{I}_{j_2}$
      vertex verification
      #pragma omp critical $(\mathbb{I})$
      $\mathbb{I} = \mathbb{I} \cup \{\mathbf{I}_{j_1}, \mathbf{I}_{j_1}\}$
    end if
  end for
end while

Algorithm 3 Parallel (MPI) branch and bound with various Lipschitz bounds

Cover feasible region $\mathbf{D}$ by $\mathbb{I} \leftarrow \{\mathbf{I}_j | \mathbf{D} = \cup \mathbf{I}_j, j = 1, \ldots, n! \}$ using face-to-face vertex triangulation
$\mathbb{I}$ evenly (if possible) divided among the p processors $\mathbb{I} = \cup \mathbb{I}^r, |\mathbb{I}^r| \approx |\mathbb{I}|/p$
$\mathbf{S}^r \leftarrow \emptyset$, $LB(\mathbb{I}^r) \leftarrow -\infty$
while $(\mathbb{I}^r \neq \emptyset)$ **do**
  Choose $\mathbf{I}_j^r \in \mathbb{I}^r$ using selection strategy, $\mathbb{I}^r \leftarrow \mathbb{I}^r \backslash \{\mathbf{I}_j^r\}$
  $LB(\mathbb{I}^r) \leftarrow \max\{LB(\mathbb{I}^r), \max\limits_{v \in V(\mathbf{I}_j^r)} f(v)\}$
  Interchange the best currently found value $LB(\mathbb{I}^r)$ among processors
  $\mathbf{S}^r \leftarrow \arg\max\{f(\mathbf{S}^r), \max\limits_{v \in V(\mathbf{I}_j^r)} f(v)\}$
  calculate $UB(\mathbf{I}_j^r)$ using different proposed Lipschitz bounds
  if $(UB(\mathbf{I}_j^r) > LB(\mathbb{I}^r) + \varepsilon)$ **then**
    Branch $\mathbf{I}_j^r$ into 2 simplices: $\mathbf{I}_{j_1}^r$, $\mathbf{I}_{j_2}^r$
    $\mathbb{I}^r = \mathbb{I}^r \cup \{\mathbf{I}_{j_1}^r, \mathbf{I}_{j_2}^r\}$
  end if
end while
Collect $\mathbf{S}^r$

- Each processor covers one part of the feasible region. This is shown symbolically $\mathbb{I} = \cup \mathbb{I}^r, |\mathbb{I}^r| \approx |\mathbb{I}|/p$ using division by the number of processors.

- The best currently found value of the objective function $LB(\mathbb{I}^r)$ is local – processors do not interchange it. (in the other MPI version processors interchange the best currently found values of the objective function whenever they are found).
- After completion the results $\mathbf{S}^r$ of the optimization are collected. The best found solution is a numerical approximation of the global solution.

3. Experimental Investigation of Lipschitz Optimization Algorithms with Simplicial Partitions

In this chapter, the results of computational experiments are presented and discussed. Various test problems ($2 \leq n \leq 6$) for global optimization have been used in our experiments. The speed of global optimization has been estimated using various criteria.

Computational comparison of various bounds

In this section, we have shown that, for dimension $n = 2$ test problems, a proposed corrected Lipschitz bound based on 2 extreme (infinite and first) norms $\mu_2^{1,\infty}$ gives by 21% smaller number of function evaluations than the bound based on the Euclidean norm μ_2^2, and, for dimension $n \geq 3$, proposed improved bound based on 3 (infinite, Euclidean and first) norms:

$$\mu_2^{1,2,\infty}(\mathbf{I}) = \min_{v \in V(\mathbf{I})} \{f(v) + K(V(\mathbf{I}))\}, \tag{6}$$

where

$$K(V(\mathbf{I})) = \min \left\{ L_\infty \max_{x \in V(\mathbf{I})} \|x - v\|_1, L_2 \max_{x \in V(\mathbf{I})} \|x - v\|_2, L_1 \max_{x \in V(\mathbf{I})} \|x - v\|_\infty \right\}$$

gives 52% smaller number of function evaluations than in the case the Euclidean norm is used alone.

In this section it has been shown, that the Piyavskii type Lipschitz bound with the first norm φ^1 gives better results for Lipschitz optimization than that used as usual μ_2^1. Depending on the dimensionality of test problems, the number of function evaluations is from 4% to 30% smaller than with a simpler bound.

The further investigation has shown that a aggregate Lipschitz bound composed of a $\mu_2^{1,2,\infty}$ and φ^1 bounds:

$$\varphi^1\mu_2^{2,\infty}(\mathbf{I}) = \min\left\{\varphi^1(\mathbf{I}), \mu_2^{2,\infty}(\mathbf{I})\right\}$$

$$= \min\left\{\max_{x\in\mathbf{I}}\left(\min_{v\in V(\mathbf{I})}\left\{f(v) + L_\infty \left\|x - v\right\|_1\right\}\right), \min_{v\in V(\mathbf{I})}\left\{f(v) + K'(V(\mathbf{I}))\right\}\right\},$$

where

$$K'(V(\mathbf{I})) = \min\left\{L_1 \max_{x\in\mathbf{I}} \left\|x - v\right\|_\infty, L_2 \max_{x\in\mathbf{I}} \left\|x - v\right\|_2\right\}$$

yield better results. On average the numbers of function evaluations are smaller by 9%, compared to $\mu_2^{1,2,\infty}$, and by 25% smaller compared to φ^1.

Computational comparison with the other well known Lipschitz optimization algorithms

The purpose of this section is to compare the proposed simplicial branch and bound algorithm with aggregate bound

$$\varphi^1\psi^2\mu_2^{2,\infty}(\mathbf{I}) = \min\left\{\varphi^1\mu_2^{2,\infty}(\mathbf{I}), \psi^2(\mathbf{I})\right\}$$

and vertex verifications $\widehat{\varphi^1\psi^2\mu_2^{2,\infty}}$ with other well-known algorithms for Lipschitz optimization, described by Hansen and Jaumard (1995). Two classes of algorithms are considered:

1. Algorithms using a single upper-bounding function, i.e. variants of Piyavskii algorithm:
   - Mladineo (Mla, 1986);
   - Jaumard, Herrmann ir Ribault (JHR, 1995);
   - Wood (Wood, 1991).
2. Branch and bound algorithms:
   - Proposed simplicial branch and bound algorithm with aggregate Lipschitz bound ($\widehat{\varphi^1\psi^2\mu_2^{2,\infty}}$);
   - Galperin (Gal85, Gal88);
   - Pinter (Pint, 1986);

Table 1. Comparison with the other well known Lipschitz optimization algorithms

No.	Mla	JHR	Wood	$\widehat{\varphi^1\psi^2\mu_2^{2,\infty}}$	Gal85	Gal88	Pint	MM	GHJ
1.	320	323	5528	**412**	3553	1713	3807	1749	643
2.	80	80	2861	**122**	1036	577	1762	744	167
3.	2066	2066	70955	**2551**	24214	16089	28417	10839	3531
4.	6	6	157	**5**	106	73	1527	94	45
5.	41	41	209	**36**	430	217	907	424	73
6.	548	548	14740	**691**	7729	2929	7772	2684	969
7.	-	5088	183759	**6240**	43123	34705	62917	22799	7969
8.	177	177	1403	**212**	2113	1289	2272	964	301
9.	-	8838	309763	**11049**	57814	49873	88932	53549	13953
10.	673	673	18613	**830**	8508	5628	9022	3814	1123
11.	1613	1613	53348	**1931**	18235	12737	20312	9224	2677
12.	-	8414	470200	**8926**	63088	56177	105572	45389	12643
13.	-	9617	-	**9855**	65536	59049	109227	35949	15695
14.	>460	>41700	-	495711	5383113	3886897	-	-	**215061**
15.	>290	9363	-	**1030**	635909	347075	-	-	24249
16.	>290	>12000	-	**536761**	15620627	-	-	-	1297205
17.	>280	>14400	-	**229049**	12481708	-	-	-	268279
18.	>690	1309	-	**3091**	46411	23765	-	-	3219
19.	446	445	-	**4684**	35463	18669	-	-	7177

- Meewella and Mayne (MM, 1988);
- Gourdin, Hansen and Joumard (GHJ, 1994).

The comparison of algorithms is based on the number of function evaluation criterion. The numbers of function evaluations are presented in Table 1.

It is mentioned in Hansen and Jaumard (1995), that the results for all algorithms may be obtained only when the required precision is not too restrictive. Even so, some problems cannot be solved by some algorithms in reasonable computational time and/or memory size. In the experiments we apply the precision used in Hansen and Jaumard (1995). The numbers of function evaluations are smallest, when the algorithms of Mladineo and of Jaumard, Herrmann and Ribault are used. However, these algorithms belong to the first class and require a longer computational time. The branch and bound algorithms require larger numbers of function evaluations, but much shorter computational time. The performance of the proposed branch and bound algorithm with aggregate bound $\widehat{\varphi^1\psi^2\mu_2^{2,\infty}}$ is similar to that of the best branch and bound algorithm (GHJ) and often it is even better.

Results of experiments with parallel algorithms

In this section, the results of the parallel computational experiments are presented and discussed. Computational experiments of both OpenMP algorithm versions were performed on the parallel machine Ness at Edinburgh Parallel Computing Center (http://www.epcc.ed.ac.uk/facilities/ness/). Up to 16 processor-cores have been used in the experiments.

The performance of both MPI algorithm versions were tested on a cluster of personal computers (Vilkas cluster at Vilnius Gediminas Technical University, http://vilkas.vgtu.lt/). Up to 16 Intel Core i7-860 cores have been used in the experiments.

The parallel versions of the algorithms were evaluated using a commonly used criteria of parallel algorithms: speedup $s_p = t_1/t_p$ and efficiency of parallelization $e_p = s_p/p$, where t_p is time used by the algorithm implemented on p processors.

Optimization of some problems takes less than a second on a single processor. It is not worthwhile to parallelize the solution of such problems. Therefore, in the parallel experiments only more difficult test problems (with a solution time on a single processor of more than 1 s) are used. The number of function evaluations is smallest for the OpenMP version with vertex verification. However, the optimization time is also largest using OpenMP version with vertex verification. The number of function evaluations and optimization time is very similar when using the OpenMP version without vertex verifications and the MPI version.

For most test problems the number of function evaluations using the MPI version with interchange of the best currently found function values are very similar compared with the MPI version without interchange. For some test problems the number of function evaluations using MPI with interchange are up to 5% smaller than that using MPI without interchange. However, the optimization time using MPI with interchange is almost always bigger than for MPI without interchange and is on average about 10% larger.

The diagrams of criteria of parallelization: speedup s_p and efficiency e_p for various numbers of processors and various dimensionality (n) of test problems are shown in Figs. 1-2. The diagrams show, that the efficiency of parallelization for all parallel versions of algorithm is better for higher dimensionality test problems ($n \geq 5$) and much better efficiency is achieved using the MPI versions. The possible reason of this is that significantly less time is spent on communications. The efficiency of parallelization decreases

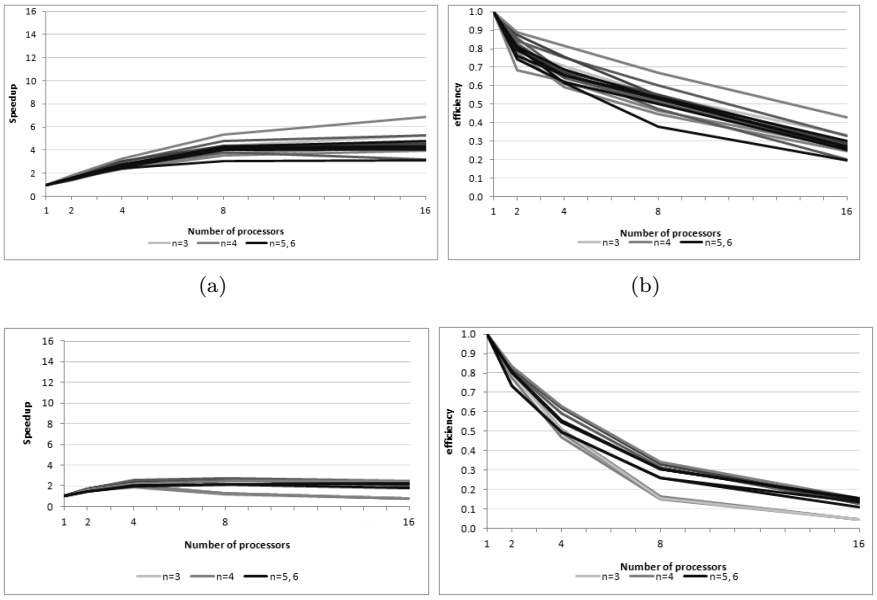


Fig 1. Speedup and efficiency of parallel OpenMP version (with vertex verification – (a), (b); without vertex verification – (c), (d))

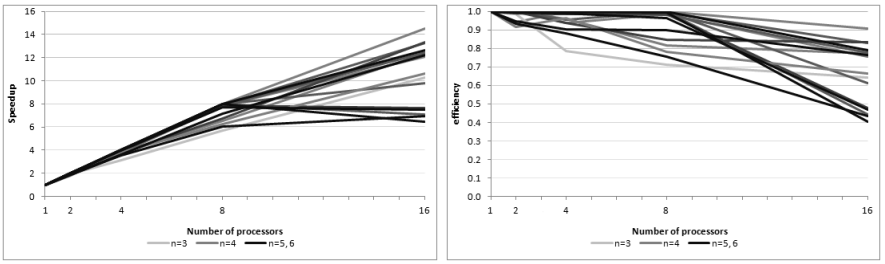
less when the number of processors is increased for difficult test problems compared with simpler test problems.

Experimental investigation of selection strategies

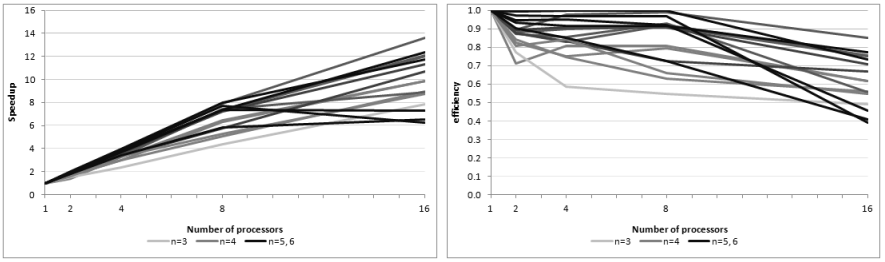
Node selection rules influence the efficiency of the branch and bound algorithm and the number of nodes kept in the candidate list. The goal of this section is to experimentally investigate the influence of selection strategies to a sequential and parallel algorithms.

The main strategies of selection are:

- Best first – select a candidate with maximal upper bound. The candidate list can be implemented using heap or priority queue.
- Depth first – select the youngest candidate. A node with the largest level in the search tree is chosen for exploration. A FILO structure is used for the candidate list which can be implemented using a stack.



(e) (f)



(g) (h)

Fig 2. Speedup and efficiency of parallel MPI version (without interchange of the best currently found function values – (a), (b); with interchange of the best currently found function values – (c), (d))

- Breadth first – select the oldest candidate. All nodes at one level of the search tree are processed before any node at the next level is selected. A FIFO structure is used for the candidate list which can be implemented using a queue.
- Statistical – In this strategy the candidate with the maximal criterion value is chosen. The candidate list can be implemented using heap or priority queue.

Results of sequential branch and bound algorithm

In this section the proposed sequential branch and bound algorithm for global optimization has been investigated. The results of different selection strategies have been compared. The average numbers of function evaluations ($\overline{f.eval}$) and average execution time ($\overline{t(s)}$) for different dimensionality test problems are shown in Table 2. For $n = 2$-dimensional

Table 2. Average numbers of function evaluations and execution time

	Best First		Depth first		Breadth first		Statistical	
n	$\overline{f.eval}$	$\overline{t(s)}$	$\overline{f.eval}$	$\overline{t(s)}$	$\overline{f.eval}$	$\overline{t(s)}$	$\overline{f.eval}$	$\overline{t(s)}$
2	6312	0.020	7161	0.020	6316	0.018	6343	0.020
3	973890	8.31	993281	7.23	974276	7.15	973909	8.15
4	1038071	36.66	1035519	35.65	1035082	35.68	1034433	36.76
5	3805578	716.35	3809086	714.39	3815658	712.42	3804724	715.10
6	1105367	1324.75	1105367	1323.92	1105367	1318.59	1105367	1374.24

Table 3. Average total numbers of simplices and average maximal size of candidate list

	Best First		Depth first		Breadth first		Statistical	
n	$\overline{TNS}$	$\overline{MCS}$	$\overline{TNS}$	$\overline{MCS}$	$\overline{TNS}$	$\overline{MCS}$	$\overline{TNS}$	$\overline{MCS}$
2	12620	2666	14320	12	12623	2359	12685	474
3	1947739	420870	1986557	23	1948111	381118	1947811	94975
4	2067422	236747	2071013	37	2069148	310379	2068842	100665
5	7609327	848062	7618053	134	7620586	719991	7609327	134498
6	4009784	188106	4012693	729	4013537	243597	4009784	106050

test problems the depth first selection strategy is the least efficient. For test problems with higher dimensionality $n \geq 3$ the average number of function evaluations are very similar for all selection strategies and the differences are insignificant. For test problems of all dimensionalities the smallest execution time is achieved when depth first and breath first selection strategies are used, despite the fact that sometimes the number of function evaluations is higher. A possible reason is that the time of insertion and deletion of elements to/from such a type of structure does not depend on the number of elements in the list. Best first and statistical selection strategies require prioritized list of candidates, and even with heap structure insertion is time consuming when the number of elements in the list is large.

The average total numbers of simplices ($\overline{TNS}$) and the average maximal sizes of candidate list ($\overline{MCL}$) are shown in Table 3. For $n = 2$-dimensional test problems (TNS) is largest when depth first selection strategy is used. For higher dimensionality $n \geq 3$ the values of (TNS) are very similar for all selection strategies. But the maximal candidate list (MCL) at the search tree varies significantly depending on selection strategies. The best results (the smallest (MCL)) achieved when depth first selection strategy is used and it is up to 7000 times smaller than (MCL) with other selection strategies. The maximal candidate list (MCL) is largest when breadth first selection strategy is used. When statistical selection strategy is used (MCL) is up to 5 times smaller than when the best first strategy is used.

Results of parallel branch and bound algorithm

In this section, the parallel MPI algorithm has been evaluated using standard criteria: speedup and efficiency of parallelization. For test problems of dimensionalities $n = 2$ and $n = 3$ the best average efficiency with various numbers of processors p is achieved when breadth first selection strategy is used. The efficiency of parallelization is very similar when best first and statistical selection strategies are used. The worst efficiency of parallelization for dimensionalities $n = 2$ and $n = 3$ is experienced when depth first selection strategy is used. For higher dimensionalities $n \geq 4$ the average parallel efficiency is similar for all selection strategies.

General Conclusions

1. More precise Lipschitz bounds of the μ_2 type which use various norms were created. The Lipschitz bounds of a new ψ^2 type based on the radius of the multidimensional circumscribed sphere and the method with which the Lipschitz bounds of the Piyavskii type are obtained when solving the linear equation systems were proposed. The aggregate bounds which use the suggested bounds with various norms were created. The results of the computational experiment with various test problems revealed that the suggested bounds were more precise than the traditional Lipschitz bounds and their use significantly decreased the number of function evaluations.
2. The sequential simplicial branch and bound algorithm that uses various proposed bounds was created. Based on it, the software for solving the Lipschitz optimization problems was developed. The experimental tests which were carried out showed the advantages of the suggested simplicial branch and bound algorithm compared to the best reviewed branch and bound algorithm of the Lipschitz optimization (GHJ).
3. The created parallel simplicial branch and bound algorithms as well as the software are intended for the shared and distributed memory computers. The efficiency of the parallel algorithm intended for the distributed memory computers is significantly higher than that of the one intended for the shared memory computers; however, the latter version of the algorithm is important because of its simple vertex verification due to which the number of function evaluations

is decreased considerably and the efficiency of parallelization of the algorithm is increased.

4. It was determined that the amount of function calculations of the parallel algorithm intended for the distributed memory computers with the best found function value interchange or without it differ only slightly. The efficiency of parallelization is larger independently from the implementation of the parallel algorithm and when the optimization problems are more complex and the number of processors is increased it decreases slower.

5. The algorithm efficiency was tested based on the criteria of number of function evaluations, optimization time, total number of elements and maximal size of candidate list. The obtained results revealed that there was no one particular selection strategy that could be used to obtain the best optimization results based on all the research criteria. Therefore, it is reasonable to select the selection strategy depending on the calculation resources of the optimization problem being solved.

List of Published Works on the Topic of the Dissertation

In the reviewed scientific periodical publications

[A1] Paulavičius, R.; Žilinskas, J. Analysis of different norms and corresponding Lipschitz constants for global optimization. *Technological and Economic Development of Economy*, ISSN 1392-8619, 12(4), 2006, p. 301–306. [Business Source Complete, Iconda]. http://www.tede.vgtu.lt/upload/ukis_zurn/07_2006_nr4.pdf

[A2] Paulavičius, R.; Žilinskas, J. Analysis of different norms and corresponding Lipschitz constants for global optimization in multidimensio case. *Information Technology and Control*, ISSN 1392-124X, 36(4), 2007, p. 383–387. [ISI Web of Science, Inspec]. http://itc.ktu.lt/itc364/Paulav364.pdf

[A3] Paulavičius, R.; Žilinskas, J. Improved Lipschitz bounds with the first norm for function values over multidimensional simplex. *Mathematical Modelling and Analysis*, ISSN 1392-6292, 13(4), 2008, p. 553–563. [ISI Web of Science, Academic Search Complete, Inspec,

Zentralblatt MATH]. doi:10.3846/1392-6292.2008.13.553-563.
http://inga.vgtu.lt/~art/k_m13_fileslist.php?key_m=1317

[A4] Paulavičius, R.; Žilinskas, J. Global optimization using the branch-and-bound algorithm with a combination of Lipschitz bounds over simplices. *Technological and Economic Development of Economy*, ISSN 1392-8619, 15(2), 2009, p. 310–325. [ISI Web of Science]. doi:10.3846/1392-8619.2009.15.310-325.
http://www.tede.vgtu.lt/en/lt/3/NR/PUB/15688

[A5] Paulavičius, R.; Žilinskas, J. Parallel branch and bound algorithm with combination of Lipschitz bounds over multidimensional simplices for multicore computers. In: R. Čiegis, D. Henty, B. Kågström, J. Žilinskas (Eds.), *Parallel Scientific Computing and Optimization. Vol. 27 of Springer Optimization and Its Applications*, Springer, ISSN 1931-6828, 2009, p. 93–102. [ISI Web of Science (Conference Proceedings Citation Index), SpringerLink, Inspec, Zentralblatt MATH]. doi:10.1007/978-0-387-09707-7_8.
http://www.springerlink.com/content/q135471381615624/

[A6] Paulavičius, R.; Žilinskas, J.; Grothey, A. Investigation of selection strategies in branch and bound algorithm with simplicial partitions and combination of Lipschitz bounds. *Optimization Letters*, ISSN 1862-4472, 4(2), 2010, p. 173–183. [ISI Web of Science, SpringerLink]. doi:10.1007/s11590-009-0156-3.
http://www.springerlink.com/content/9714284487524027/

In the other editions

[A7] Paulavičius, R.; Žilinskas, J. Branch and bound with simplicial partitions and combination of Lipschitz bounds for global optimization. In: L. Sakalauskas, G.W. Weber, E.K. Zavadskas (Eds.), *The 20th International Conference EURO Mini Conference Continuous Optimization and Knowledge-Based Technologies (EurOPT-2008), May 20-23, 2008, Neringa, Lithuania*, ISBN 978-9955-28-283-9, 2008, p. 54–58. [ISI Web of Science (Conference Proceedings Citation Index)].
http://www.vgtu.lt/upload/leid_konf/paula_54-58.pdf

[A8] Paulavičius, R. Parallel multidimensional Lipschitz optimization. *Science and Supercomputing in Europe. HPC-Europe Report.* 2008, p. 257–261.

About the author

Remigijus Paulavičius was born on the 26th of Januar, 1982 in Ignalina district, Didžiasalis. First degree in Mathematics, Faculty of Mathematics and Informatics, Vilnius Pedagogical University, 2004. Master of Science in Mathematics, Faculty of Mathematics and Informatics, Vilnius Pedagogical University, 2006. In 2006–2010 – PhD student of Institute of Mathematics and Informatics. In 2004–2006 was working as a technician, and in 2006–2010 – as an assistant at the Vilnius Pedagogical University, Faculty of Mathematics and Informatics. At present – lector at the Department of Algebra and Statistics, Faculty of Mathematics and Informatics, Vilnius Pedagogical University. Remigijus Paulavičius worked on probation at EPCC (the supercomputing centre at The University of Edinburgh), UK.

GLOBALUSIS OPTIMIZAVIMAS SU SIMPLEKSINIAIS POSRIČIAIS

Tyrimų sritis

Globaliojo optimizavimo metodai naudojami skaičiuojamosios chemijos ir biologijos, biomedicinos, operacijų tyrimo, ekonomikos, inžinerinio projektavimo ir valdymo bei daugelio kitų inžinerijos ir taikomųjų mokslų uždavinių sprendimui. Viena iš labiausiai tyrinėjamų globaliojo optimizavimo sričių yra Lipšico optimizavimas, kuris remiasi prielaida, kad tikslo funkcija tenkina Lipšico sąlygą. Daugiamačiai Lipšico optimizavimo metodai tiek teorine, tiek ir praktine prasme yra gerokai mažiau ištirti nei vienmačiai. Šie metodai dažniausiai naudoja stačiakampius posričius ir paprastai apskaičiuojamus Lipšico rėžius, o simpleksiniams posričiams ir rėžių gerinimui skirtas mažesnis dėmesys. Todėl šio darbo tyrimų sritis yra daugiamačiai simpleksiniai Lipšico optimizavimo algoritmai, naudojantys agreguotuosius Lipšico rėžius su įvairiomis normomis.

Darbo aktualumas

Sparčiai besivystančios informacinės technologijos ir tobulėjanti kompiuterinė technika leidžia spręsti vis sudėtingesnius optimizavimo uždavinius, kurių išsprendžiamumas dar visai neseniai buvo sunkiai arba iš viso neįmanomas. Globaliojo optimizavimo metodai, pagrįsti Lipšico rėžių apskaičiavimu, yra pakankamai plačiai išnagrinėti ir yra plačiai taikomi įvairių optimizavimo uždavinių sprendimui. Šių metodų pritaikomumą

lemia tai, kad jie pateikia tą patį atsakymą kiekvieną kartą, skirtingai negu stochastiniai optimizavimo metodai. Lipšico metodai garantuoja surasto sprendinio tikslumą per baigtinį (gali būti ir ilgas) laiką, priešingai negu tiesioginės paieškos ar stochastiniai metodai. Šie metodai gali būti naudojami „juodosios dėžės“ situacijoje, skirtingai nuo intervalų metodų.

Tačiau Lipšico optimizavimo metodai turi ir svarbių trūkumų, lyginant su kitais globaliojo optimizavimo metodais. Lipšico metodai dažniausiai remiasi prielaida, kad Lipšico konstanta žinoma iš anksto, o tai retas atvejis sprendžiant praktinius uždavinius. Lipšico algoritmai yra lėtesni negu kai kurie kiti globaliojo optimizavimo algoritmai. Didėjant matmenų skaičiui n, Lipšico algoritmų skaičiavimo apimtys didėja eksponentiškai.

Dėl to geresnių rėžių, lemiančių greitesnius Lipšico algoritmus, kūrimas užima pagrindinę disertacijos dalį. Svarbią disertacijos dalį užima pasiūlytų algoritmų efektyvių lygiagrečiųjų versijų realizavimas ir įvertinimas, siekiant spręsti didesnius optimizavimo uždavinius.

Tyrimų objektas

Disertacijos tyrimų objektas yra:

- Lipšico globaliojo optimizavimo algoritmai daugelio kintamųjų erdvėse, naudojantys simpleksinius posričius;
- lygiagrečiųjų kompiuterių sistemų taikymas globaliajam optimizavimui.

Darbo tikslas ir uždaviniai

Pagrindinis šio darbo tikslas – modifikuoti esamus Lipšico globaliojo optimizavimo algoritmus daugelio kintamųjų erdvėse ir pasiūlyti naujus siekiant greičiau ir tiksliau spręsti optimizavimo uždavinius, bei panaudojant lygiagrečiųjų kompiuterių sistemas spręsti didesnius optimizavimo uždavinius. Siekiant užsibrėžto tikslo buvo sprendžiami šie uždaviniai:

1. Apžvelgti esamus globaliojo optimizavimo algoritmus ir apsibrėžti tiriamų algoritmų grupę;
2. Ištirti, kaip optimizavimo rezultatus įtakoja Lipšico rėžių sudarymui naudojamos įvairios normos ir jas atitinkančios Lipšico konstantos.
3. Naudojant simpleksinius posričius sukonstruoti:
   - patikslintus įprastus Lipšico rėžius, naudojančius įvairias normas ir jas atitinkančias Lipšico konstantas.

- Pijavskij tipo Lipšico rėžius, daugiamačiu atveju gaunamus sprendžiant tiesinių lygčių sistemas.
- Lipšico rėžius, pagrįstus apibrėžtinės daugiamatės sferos spinduliu.
- agreguotuosius Lipšico rėžius sudarytus iš kelių skirtingų Lipšico rėžių tipų su įvairiomis normomis.

4. Sukurti nuosekliuosius ir lygiagrečiuosius Lipšico optimizavimo simpleksinius šakų ir rėžių algoritmus su pasiūlytais rėžiais.
5. Palyginti sukurtų algoritmų efektyvumą naudojant įvairius pasiūlytus Lipšico rėžius.
6. Gautus rezultatus palyginti su kitais gerai žinomais Lipšico optimizavimo algoritmais.
7. Ištirti sukurtų nuosekliųjų ir lygiagrečiųjų algoritmų efektyvumą naudojant įvairias paieškos strategijas: geryn, platyn, gilyn ir statistinę.
8. Palyginti sukurtų lygiagrečiųjų algoritmų, skirtų bendrosios ir paskirstytosios atminties kompiuteriams, lygiagretinimo efektyvumą.

Mokslinis darbo naujumas

Remiantis teoriniais ir eksperimentiniais tyrimais, ištirta, kaip Lipšico optimizavimo rezultatus įtakoja naudojamos normos ir jas atitinkančios Lipšico konstantos. Gautų rezultatų pagrindu pasiūlyta įprastų Lipšico rėžių skaičiavimui vietoj vienos konkrečios normos naudoti kelias bei apjungti skirtingų tipų rėžius su įvairiomis normomis. Tokiu būdu gaunami tikslesni agreguotieji Lipšico rėžiai neskaičiuojant tikslo funkcijos papildomuose taškuose.

Naudojant pirmąją normą pasiūlyta, kaip Pijavskij tipo rėžius apskaičiuoti sprendžiant tiesinių lygčių sistemas vietoj kvadratinių Euklidinės normos atveju.

Euklidinės normos atvejui, pasiūlytas naujas Lipšico rėžis, pagrįstas apibrėžtinės daugiamatės sferos spinduliu.

Pasiūlyto simpleksinio šakų ir rėžių algoritmo efektyvumas ištirtas naudojant įvairius pasiūlytus Lipšico rėžius ir palygintas su kitais gerai žinomais Lipšico algoritmais.

Pagal funkcijų skaičiavimo kiekio, optimizavimo laiko, ištirtų posričių skaičiaus bei maksimalaus kandidatų sąrašo kriterijus algoritmo efektyvumas ištirtas priklausomai nuo naudojamos paieškos strategijos: geryn, platyn, gilyn ir statistinės.

Sukurtos pasiūlyto algoritmo lygiagrečiosios realizacijos bendrosios (naudojant OpenMP standartą) ir paskirstytosios (naudojant MPI standartą) atminties kompiuteriams.

Sukurtų algoritmų pagrindu realizuota programinė įranga:

- nuoseklioji versija: http://www.gridglobopt.vgtu.lt/files/AT_Projekto_Ataskaita/BranchBoundTemplate/BBSequential/Examples/SimplicialBBSequential/
- lygiagrečioji versija, skirta bendrosios atminties kompiuteriams: http://www.gridglobopt.vgtu.lt/files/AT_Projekto_Ataskaita/BranchBoundTemplate/BBParallel/Examples/SimplicialBBOpenMP/
- lygiagrečioji versija, skirta paskirstytosios atminties kompiuteriams: http://www.gridglobopt.vgtu.lt/files/AT_Projekto_Ataskaita/BranchBoundTemplate/BBParallel/Examples/SimplicialBBParallel/

Ginamieji disertacijos teiginiai

1. Naudojant agreguotuosius Lipšico rėžius, sudarytus iš kelių skirtingo tipo rėžių su įvairiomis normomis, gaunami tikslesni rėžiai, kurių dėka pagreitinami optimizavimo algoritmai.
2. Pijavskij tipo rėžių skaičiavimas, naudojant pirmąją normą, leidžia kvadratinių lygčių sistemų sprendimą pakeisti tiesinių lygčių sistemų sprendimu.
3. Lipšico rėžiai, pagrįsti apibrėžtinės daugiamatės sferos spinduliu, dažniausiai yra geresni už įprastai naudojamus šakų ir rėžių algoritmuose.
4. Nėra vienos konkrečios paieškos strategijos, kurią naudojant būtų gaunami geriausi optimizavimo rezultatai pagal visus tirtus kriterijus.

5. Atlikus viršūnių patikrinimą stipriai sumažėja tikslo funkcijų skaičiavimų kiekis bei padidėja lygiagrečiojo bendrosios atminties kompiuteriams skirto algoritmo efektyvumas.

6. Paskirstytosios atminties kompiuteriams skirto lygiagrečiojo algoritmo funkcijų skaičiavimų kiekis su geriausios surastos funkcijos reikšmės apsikeitimu ir be jo, skiriasi nežymiai.

Darbo apimtis

Disertaciją sudaro įvadas, trys skyriai ir bendrosios išvados. Papildomai disertacijoje yra pateikta: naudotų žymėjimų ir santrumpų sąrašas, eksperimentiniuose tyrimuose naudoti optimizavimo uždaviniai, sąvokų žodynas, dalykinė rodyklė bei literatūros sąrašas. Bendra disertacijos apimtis yra 134 puslapiai, kuriuose pateikta: 20 paveikslų, 15 lentelių ir 5 algoritmai. Disertacijoje remtasi 135 literatūros šaltiniais.

Bendrosios išvados

1. Sukonstruoti tikslesni μ_2 tipo Lipšico rėžiai, naudojantys įvairias normas. Pasiūlyti nauji ψ^2 tipo Lipšico rėžiai, pagrįsti daugiamatės apibrėžtinės sferos spinduliu. Pasiūlytas metodas, kuriuo Pijavskij tipo Lipšico rėžiai gaunami sprendžiant tiesinių lygčių sistemas. Sudaryti agreguotieji rėžiai, naudojantys pasiūlytus rėžius su įvairiomis normomis. Skaičiuojamojo eksperimento rezultatai su įvairiais testiniais uždaviniais parodė, kad pasiūlyti rėžiai yra tikslesni už tradicinius Lipšico rėžius, o juos naudojant stipriai sumažėja tikslo funkcijų skaičiavimų kiekis.

2. Sudarytas nuoseklusis simpleksinis šakų ir rėžių algoritmas naudojantis įvairius pasiūlytus rėžius. Jo pagrindu realizuota programinė įranga Lipšico optimizavimo uždaviniams spręsti. Atlikti eksperimentiniai tyrimai parodė pasiūlytojo simpleksinio šakų ir rėžių algoritmo pranašumą lyginant jį su geriausiu apžvelgtu šakų ir rėžių Lipšico optimizavimo algoritmu (GHJ).

3. Sudaryti lygiagretieji simpleksiniai šakų ir rėžių algoritmai bei programinė įranga skirta bendrosios ir paskirstytosios atminties kompiuteriams. Paskirstytosios atminties kompiuteriams skirto lygiagrečiojo algoritmo efektyvumas yra žymiai geresnis, nei bendrosios atminties kompiuteriams, tačiau pastaroji algoritmo versija svarbi dėl nesudėtingos viršūnių patikrinimo realizacijos, kurios dėka stipriai sumažinamas funkcijų skaičiavimų kiekis bei padidinamas algoritmo lygiagretinimo efektyvumas.

4. Ištirta, kad lygiagrečiojo paskirstytosios atminties kompiuteriams skirto algoritmo funkcijų skaičiavimų kiekis su geriausios surastos funkcijos reikšmės apsikeitimu ir be jo, skyriasi nežymiai. Nepriklausomai nuo lygiagrečiojo algoritmo realizacijos, lygiagretinimo efektyvumas yra didesnis, o didėjant procesorių skaičiui mažėja lėčiau, kai optimizavimo uždaviniai yra sudėtingesni.

5. Pagal funkcijų skaičiavimo kiekio, optimizavimo laiko, ištirtų posričių skaičiaus bei maksimalaus kandidatų sąrašo kriterijus atlikti šakų ir rėžių algoritmų efektyvumo tyrimai parodė, kad nėra vienos konkrečios paieškos strategijos, pagal kurią būtų gaunami geriausi optimizavimo rezultatai pagal visus kriterijus, todėl paieškos strategiją tikslinga pasirinkti priklausomai nuo sprendžiamo optimizavimo uždavinio skaičiavimo resursų.

Trumpos žinios apie autorių

Remigijus Paulavičius gimė 1982 m. sausio 26 d. Ignalinos r., Didžiasalyje.

2004 m. įgijo matematikos bakalauro laipsnį Vilniaus pedagoginio universiteto Matematikos ir informatikos fakultete. 2006 m. įgijo matematikos mokslo magistro laipsnį Vilniaus pedagoginio universiteto Matematikos ir informatikos fakultete. 2006–2010 m. – Matematikos ir informatikos instituto doktorantas. 2004–2006 m. dirbo vyr. laborantu, o 2006–2010 m. – asistentu Vilniaus pedagoginio universiteto Matematikos ir informatikos fakultete. Šiuo metu dirba lektoriumi Vilniaus pedagoginio universiteto Algebros ir statistikos katedroje. Remigijus Paulavičius stažavosi superskaičiavimų centre EPCC (The Edinburgh Parallel Computing Centre), Edinburge, Jungtinėje Karalystėje.

Remigijus PAULAVIČIUS

GLOBAL OPTIMIZATION WITH SIMPLICIAL PARTITIONS

Summary of Doctoral Dissertation
Physical Sciences (P 000)
Informatics (09 P)
Informatics, Systems Theory (P 175)

Remigijus PAULAVIČIUS

GLOBALUSIS OPTIMIZAVIMAS SU SIMPLEKSINIAIS POSRIČIAIS

Daktaro disertacijos santrauka
Fiziniai mokslai (P 000)
Informatika (09 P)
Informatika, sistemų teorija (P 175)

2010 08 17, 2 sp. l. Tiražas 60 egz.
Parengė spaudai ir išleido Matematikos ir informatikos institutas
Akademijos g. 4, LT-08663 Vilnius.
Interneto svetainė: http://www.mii.lt
Spausdino „Kauno technologijos universiteto spaustuvė“
Studentų g. 54, LT-51424 Kaunas