Global Optimization by Means of Branch and Bound with Simplex Based Covering

Jens Clausen* Antanas Žilinskas †‡

December, 1998

Abstract

We consider the problem of optimizing a Lipshitzian function. A new version of the Branch and Bound technique, based on covering by simplexes, is proposed. The convergence of the algorithm is proved. A two dimensional version is implemented, and numerical results are presented.

1 Introduction

One of the most general assumptions enabling the costruction of algorithms for global optimization guaranteeing prescribed accuracy is the assumption that an objective function has bounded rate of changing and is defined over a bounded domain. In the following, we restrict our attention to functions of this type. Hence, we assume that $f(x), x \in A \subset R^n$ is Lipshitzian with constant L, and that A is bounded, i.e. that $\mid f(x) - f(y) \mid \leq L \mid\mid x - y \mid\mid$ and $\mid\mid x \mid\mid \leq B < \infty$, $x, y \in A$, where $\mid\mid \cdot \mid\mid$ denotes the Euclidean norm.

It is the obvious that if $\cup_{i=1}^{N} Sph(x_i, \epsilon/L) \supseteq A$ (with $Sph(x, r)$ being the sphere with center x and radius r), then $\min_{1 \leq i \leq N} f(x_i)$ estimates the global minimum with accuracy ϵ. The uniform distribution of N points is worst case optimal algorithm not only in the class of passive but also in the class of sequential algorithms [12]. Although $N(\epsilon)$, the number of points needed for the optimal algorithm to guarantee accuracy ϵ, is an exponential function of the dimension of the solution space, n, this poses no more pessimism regarding solution methods than in the case of an NP complete combinatorial problem. Algorithms based on the Lipshitzian model have been investigated by many authors: W.Baritompa, Y.Evtushenko, E.Galperin, E.Hansen, P.Hansen, R.Horst, B.Jaumard, J.Pinter, P.Pardalos, H.Tuy, G.Wood and others. The recent review of results in this direction of global optimization is presented in [6]. Our research was motivated by the following observations:

*Technical University of Denmark, Department of Mathematical Modeling, Building 321, DK-2800 Lyngby, Denmark. e-mail: jc@imm.dtu.dk

†Vytautas Magnus University, Institute of Mathematics and Informatics, Akademijos str. 4, Vilnius, LT-2600, Lithuania. e-mail: antanasz@ktl.mii.lt

‡This work was carried out while the second author was visiting professor at DTU supported by the Danish Natural Science Research Council.

- In practice there exist problems of modest dimensionality whose solution with guaranteed accuracy is important at the cost of any reasonable time,
- normally the objective functions are very different from the saw-tooth worst case functions,
- investigation of algorithms with guaranteed accuracy may give rise to heuristics which are efficient for some types of objective functions although not guaranteed to solve any Lipshitz problem with prescribed accuracy,
- parallel implementation of algorithms remarkably extended practical solvability in some classes of hard optimization problems [7],[4],[9].

Covering by spheres is mainly used as an abstract method for theoretical investigation. The constructive algorithms are defined by minimization of lower bounds of function values or by covering of the feasible region by hyperrectangles [6]. The disadvantages of both approaches are discussed and a new method of branch and bound search using a covering of the feasible region by simplexes is proposed. The implementation of the algorithm in the two dimensional case is discussed in detail and numerical results of the implemented algorithm are presented.

2 A new version of the Branch and Bound technique

Two main approaches are used to construct global minimization algorithms for Lipshitz continuous functions. The algorithms implementing the first approach define the point of $f(\cdot)$ to be evaluated next by global minimization of the lower bound of values of $f(\cdot)$. This technique aims for the maximal possible improvement at the current step. Experimental testing shows rather high efficiency of these algorithms for several one dimensional and two dimensional test functions with respect to number of function evaluations [6], [12]. However, even in the two dimensional case, the auxiliary problem (minimization of the lower bound) often becomes practically unsolvable [6].

The second approach is based on the covering of A by elementary subsets and iterative refinement of the cover by subdivision of a selected subset into smaller ones. A branch and bound technique controls the process of refinement. For each elementary subset, a lower bound estimate is calculated using only the values of $f(\cdot)$ at (normally few) points of this subset. If the estimate is worse than the known upper bound for $\min_{x\in A} f(x)$, then this subset may be left out from further consideration, i.e. *discarded or fathomed.* If this is not the case, the elementary subset under consideration is subdivided into several smaller subsets of the same type, i.e. *branching* takes place. Different types of elementary subsets may be used as well as different strategies for selecting the next subset to process and for subdivision of a subset. Several implementations of this technique are discussed in [6].

It is interesting to note, that some one dimensional algorithms, e.g. Pijavskij-Shubert [6], and Brent [3] may be considered as representatives of both approaches. The testing in [6] shows that for two dimensional functions, the best implementation of the branch and bound technique requires twice as many evaluations of $f(\cdot)$ as the algorithms using

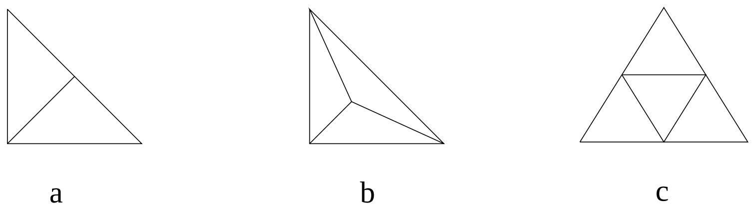

Figure 1: Three possible subdivisions of a simplex.

the first approach. But the auxiliary computations of branch and bound algorithms are essentially less time consuming. Some problems, which were too hard for algorithms using the first approach, have been solved by branch and bound algorithms.

The elementary covering subsets are normally hyperrectangles (boxes). In case interval methods are used, boxes are very natural because they correspond to the definition of operations in interval arithmetics [7], [5]. The use of boxes for other covering algorithms has serious disadvantages. For example, the use of symmetric boxes (hypercubes) is normally out of question because of the exponential number (with respect to n) of descendants occurring when dividing a parent hypercube into smaller ones. Several schems for dividing of a box into, e.g. two, smaller boxes have been proposed, however, the implied algorithms were rather slow in computational experiments [6].

Our idea is to construct a branch and bound algorithm based on potentially tight lower bounds evaluated by not too complicated auxiliary computations. The idea to choose the **regular simplexes** as elementary not overlapping covering subsets seems prospective. Let us note, that different simplex based covering techniques have been proved advantageous in [11], [15], [14], [1], [10], [8]. It is demonstrated in the current paper that Lipshitzean bound calculation for regular simplexes is a rather easy task.

The number of standard descendant simplexes obtained by dividing a parent simplex is equal to $n+2$, i.e. it is approximately equal to the dimensionality of a problem. A simplex in R^n is cut into $n+2$ simplexes by means of $n+1$ hyperplanes parallel to the facets of the parent simplex and dividing the corresponding edges in the ratio 1:1. The vertexes of $n+1$ of the descendant simplexes are one vertex of the parent simplex and n centers of the facets, whose intersection defines the relevant vertex of the parent simplex. The vertices of the $(n+2)$th descendant simplex coincide with the centers of facets of the parent simplex. The 2-dimensional case is illustrated in Figure 1 (c).

Other schemes of subdivision exist, cf. Figure 1 (a) and (b). The main advantage over these is that bounds calculated using the Lipshitz constant and the function values of the vertices of a regular simplex covering a given volume will in general be better than those calculated from the vertices of an irregular simplex covering the same volume.

The proposed subdivision leads to $n+2$ new simplexes and the evaluation of $n+1$ new points while a radial subdivision from the center of the simplex in question (cf. Figure 1 (b)) leads to $n+1$ new simplexes and only one new function evaluation. The ratio $n+1/n+2$ against $1/n+1$ suggests that the efficiency of a search algorithm based on radial subdivision may after all be better. The maximal distance from a point to a vertex resulting if radial cuts are used may, however, differ only marginally from the corresponding distance in the parent simplex, and therefore the bounds of the new simplexes may turn out to be only marginally better than that of the parent simplex. Hence, we expect radial subdivision to be less efficient than regular subdivision, and the results in [10] supports this claim.

Let $S(z,d) \subset R^n$ be a simplex with center z and edge length d. Let a set of simplexes covering A be given, i.e. $\bigcup_{i=1}^{m} S(z_i,d_i) \supseteq A, : \mu(S(z_i,d_i) \cap S(z_j,d_j)) = 0, i \neq j$, where $\mu(\cdot)$ denotes the volume on R^n. Simply choosing a suitable simplex S circumscribing A may be too inefficient, e.g. the volume ratio between A and S is less than $\left(2/\sqrt{n\,(n+1)}\right)^n$ for the hypercubic regions. We propose a possible covering for 2-dimensional rectangular domains in Section 5. A set of standard coverings should be developed for standard domains as e.g. hyperrectangels, but the design of an efficient initial covering for a complicated domain of a practical problem will in general constitute a challenge.

In each $S(z_i,d_i)$, at least one point x_j is chosen for calculation of the value of $f(\cdot)$. Using the available values of $f(\cdot)$ for each $S(z_i,d_i)$, a lower bound b_i for the function values of $f(\cdot)$ over $S(z_i,d_i)$ is calculated using the Lipshitz constant. Let y_{0k} be an estimate (an upper bound) on the global minimum after k evaluations of $f(\cdot)$, i.e. $y_{0k} \geq \min_{x\in A} f(x) : k \geq m$ (normally, the equality $y_{0k} = \min_{1\leq i\leq k} f(x_i)$ holds). The simplex $S(z_i,d_i)$ is discarded if $b_i \geq y_{0k} - \epsilon$. The remaining simplexes are kept in a priority queue ordered by the values of bounds b_i.

At each iteration of a branch and bound algorithm, the simplex $S(z_i,d_i)$ from front of the queue is chosen to be divided into $n+2$ simplexes with edge length $d_i/2$. The function values of $f(\cdot)$ are calculated at the points of $S(z_i,d_i)$ defined by the algorithm, and y_{0k} is updated. The bounds for the descendant simplexes are estimated and each simplex is discarded or inserted in the priority queue depending on the relation of its bound value to y_{0k}. The algorithm terminates when the priority queue is empty. The branch and bound algorithm is hence an eager best-first algorithm.

Two versions of the algorithm are considered:

- *Version 1.* The values of $f(\cdot)$ are calculated at the center of each simplex.
- *Version 2.* The values of $f(\cdot)$ are calculated at the vertexes of the initial simplexes. After the division of a simplex, the new function values are calculated at all or some of $n+1$ vertexes of descendant simplexes depending whether they coincide with the vertexes of previously generated simplexes (in which case the function values are already known).

In the first version of the algorithm the estimation of bounds is very simple:

$$b_i = f(z_i) - L \cdot \delta_i,$$

where δ_i denotes the distance between the center and the vertexes of the ith simplex. In the second version, b_i is the solution to the minimization problem

$$b_i = \min_{x\in S_i(z_i,d_i)} \quad \max_{x_j\in S_i(z_i,d_i)} (f(x_j) - L \cdot ||x - x_j||), \tag{1}$$

where $x_j \in S_i(z_i,d_i)$ are the trial points corresponding to the vertexes of ith simplex. It is shown in the following, that the problem (1) is easy, at least for a two dimensional case.

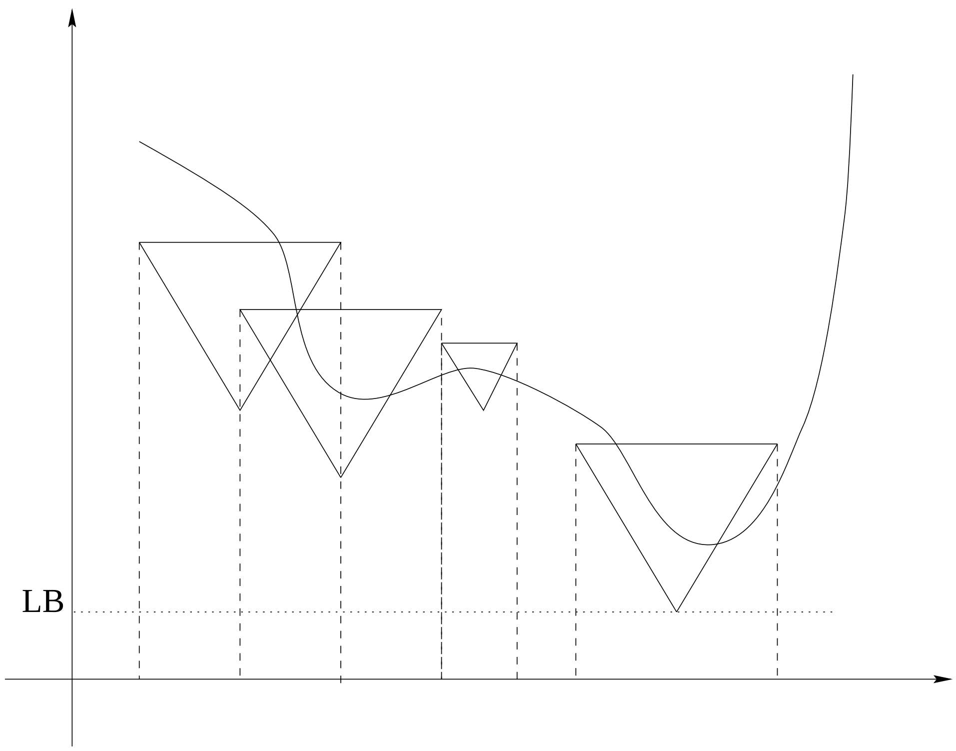

Figure 2: The $n+1$st coordinate of the apex constitutes a lower bound.

We now briefly describe the multi-dimensional bisection method of Wood in the context of branch and bound and contrast it to our branch and bound algorithm. For a more detailed description, the reader is referred to [2], [15], [14], [1].

The idea of multi-dimensional bisection (MB) is to maintain a collection of $n+1$ dimensional regular simplexes, whose union contains all points of the epigraph of f with minimal function value. Each simplex has a unique apex as the point with lowest value of the $n+1$ st coordinate, an axis parallel to the $n+1$ coordinate axis, and a top parallel to the domain space in question. Hence, for each simplex the $n+1$st coordinate of the apex provides a lower bound for the function values for all points of the domain of f located in the projection of top onto the domain space, cf. Figure 2.

In each iteration of MB a simplex is chosen for processing. The function value of the domain point corresponding to the lowest apex is computed, and based on this function value and the Lipshitz constant, a simplical cone is generated, in which no points of the epigraph of f can be located. This is called the removal cone. Three possibilities now exist: If the removal cone contains the simplex, the simplex is fathomed (just as in ordinary branch and bound). Otherwise, some parts of the simplex falls outside the removal cone, and these are either disjoint simplexes with tops being disjoint subsets of the top of the originating simplex or can be described as the union of simplexes each of which is a subsimplex of the originating simplex, cf. Figure 3 (adapted from [15]).

In both cases some of the (up to) $n+1$ simplexes are added to the collection of simplexes to be processed, namely those for which the $n+1$ coordinate of the apex is not larger than or equal to the minimum value of f in the previously evaluated points. Also,

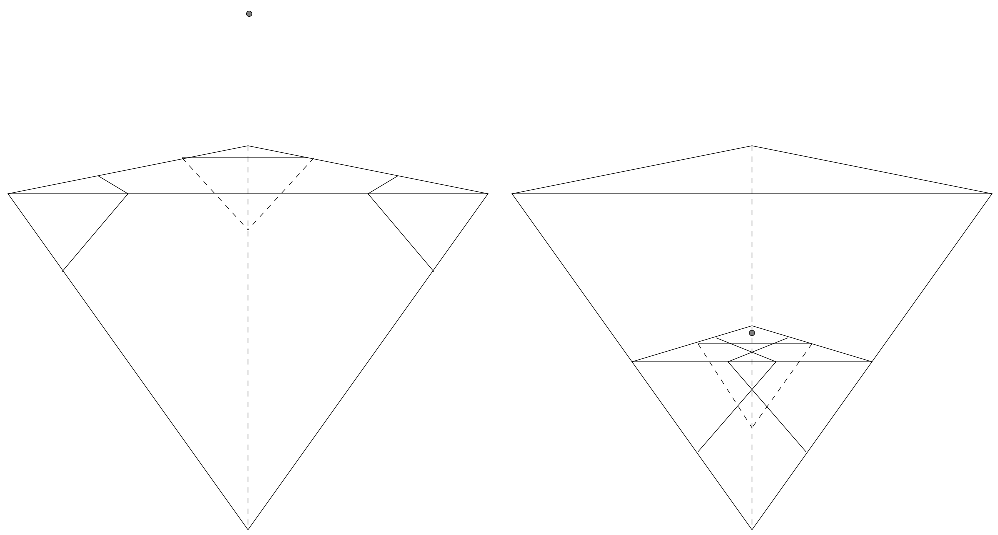

Figure 3: Subdividing the feasible region in Multidimensional Bisection.

the tops of the simplexes in the collection may be lowered if the calculated function value is the best value of f found so far.

In terms of branch and bound, the projections of the tops of the simplexes in the current collection onto the domain space constitute the live subspaces of the domain, and values of the $n+1$ coordinates of their apexes constitute the bounds of these subspaces. The function evaluation for the apex of a simplex is connected to the branching process, i.e. used to generatenew subspaces, and implicitly also used to fathom complete subspaces or parts hereof. The new resulting subspaces may overlap and will normally be of differing size.

In contrast, the first version of the branch and bound method (with the simple bound calculation based on the evaluation of f in the center of the simplex in question) presented earlier uses the function evaluations explicitly to calculate bounds, and the partition of a simplex is done in a regular fashion using no information related to the function evaluation. Hence, even if the MB method can be phrased in a branch and bound context with simplexes used as subspaces, the method differs substantially from both versions of our branch and bound algorithm. In particular, note that the function evaluations of our methods are used explicitly for bound calculations, and that our second bound calculation can be expected to produce bounds substantially stronger than the apex-bounds of the MB-method.

3 Construction of a simplex

We define the vertexes of the standard simplex with edge length equal to one iteratively. The $k+1$ vertices in dimension k are denoted χ_1^k, $\chi_2^k, \ldots, \chi_{k+1}^k$, and the center $\chi_c^k = \frac{1}{k+1}\sum_{i=1}^{k+1}\chi_i$.

- Dimension 1. The two vertices are $\chi_1^1 = 0$ and $\chi_2^1 = 1$. The center is $\chi_c^1 = 0.5$.

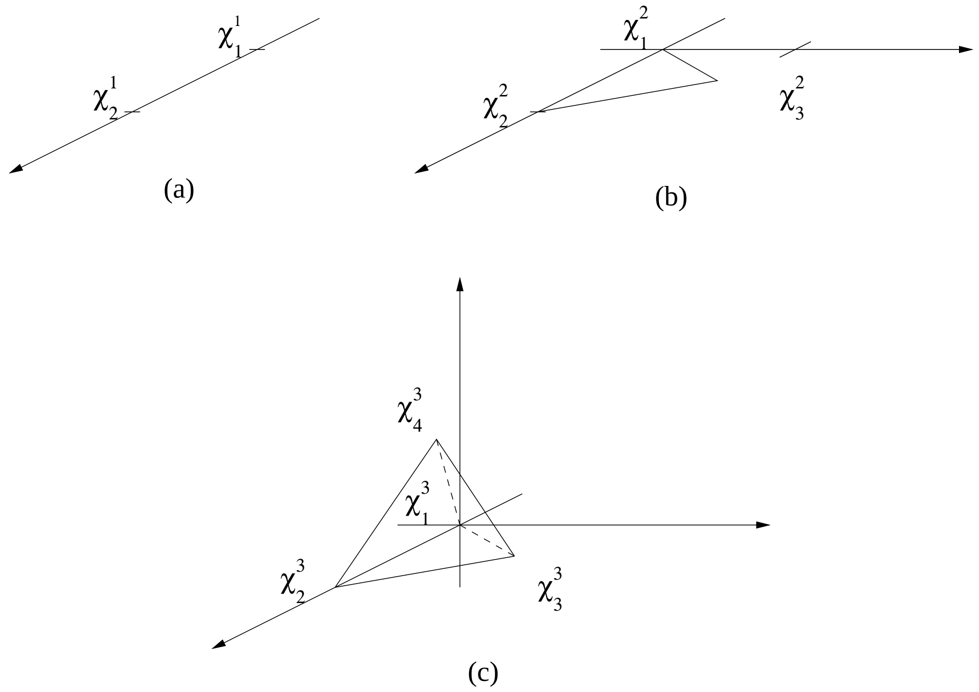

Figure 4: The simplices of dimension 1, 2 and 3.

- Dimension k. The first k vertices are constructed by locating the k vertices of the simplex of dimension $k-1$ according to their coordinates in the hyperplane defined by $x_k = 0$. The vertex χ^k_{k+1} has the first k coordinates equal to those of χ^k_c and the $k+1$ st coordinate is determined by the equality $||\chi^k_{k+1} - \chi^k_1|| = 1$.

It follows immediately that $\chi^2_1 = (0,0)$, $\chi^2_2 = (1,0)$, and $\chi^2_3 = (1/2, \sqrt{3}/2)$; in three dimensions we have $\chi^3_1 = (0,0,0)$, $\chi^3_2 = (1,0,0)$, $\chi^3_3 = (1/2, \sqrt{3}/2, 0)$, and $\chi^3_4 = (1/2, \sqrt{3}/6, \sqrt{6}/3)$, i.e. the one dimensional standard simplex is the unit interval, the two dimensional simplex is the correct triangle etc. The simplexes in dimensions 1, 2 and 3 are shown in Figure 4 (a), (b), and (c).

The general solution of the recurrence relation gives

$$\chi^k_{k+1:i} = \chi^{k-1}_{c:i},\ \chi^k_{k+1:k} = \sqrt{(k+1)/2k},\ \chi^k_{c:i} = \chi^{k-1}_{c:i},$$

$$\chi^k_{c:k} = 1/\sqrt{2k(k+1)},\ i = 1, \ldots, k-1. \tag{2}$$

All correct simplexes generated by the algorithm may be obtained from a standard simplex by contraction/extension of R^n equally in all coordinates and orthogonal transformation of R^n.

4 Convergence

The conditions of general theorems on convergence given in [11] may be checked for the proposed versions of the algorithm. However, both versions may be treated uniformly by an extremely short direct proof, which gives also some quantitatives estimates. Let us denote the number of simplexes of the initial cover by N_0 and the longest edge of these simplexes by d_{max}.

Theorem 1 *The requested accuracy ϵ is achieved by the algorithm after at most $N = (n+1) \cdot N_0 \cdot \lceil \log(d_{max}L/\epsilon) \rceil$ calculations of objective function values:*

$$y_{0N} = \min_{1 \le i \le N} f(x_i) \le \min_{x \in A} f(x) + \epsilon.$$

Proof: After $N_C = \lceil \log(d_{max}L/\epsilon) \rceil$ cuts of the largest simplex (let its number be h), the edge will become shorter than ϵ/L implying $b_h \ge \min_{x_i \in S(z_h, d_h} f(x_i) - \epsilon$. Termination now appears after $N_0 \cdot N_C \cdot (n+1)$ steps since then all generated simplexes have been discarded. The accuracy of evaluation of the global minimum is bounded by ϵ by the rule for discarding a simplex.

Theorem 2 *If the algorithm runs with $\epsilon = 0$, then only the points of global minimum are the accumulating points of the sequence x_i.*

Proof. Let $z \in A$ be a point not belonging to the set of global minimum points. Let us denote $\theta = f(z) - \min_{x \in A} f(x)$. The simplex generated by the algorithm $S(z_i, d_i) \ni z$ will be discarded after at most $N = (n+1) \cdot N_0 \cdot \lceil \log(3d_{max}L/\theta) \rceil$ calculations of objective function values, because for the hth simplex with edge length $\theta/3L$ it holds that

$$b_h \ge \min_{x_i \in S(z_h, d_h)} f(x_i) - \theta/3 \ge f(z) - 2\theta/3 = \min_{x \in A} f(x) + \theta/3 \ge y_{0N}.$$

Therefore z can not be an accumulating point of the sequence x_i.

5 Two dimensional implementation

The feasible region is supposed to be rectangular. We propose a possible initial cover, which is only slightly larger than the feasible region. The construction of the cover is rather ad hoc, but in pilot experiments the cover proved to be efficient. Of course, other covers are possible. In the solution of combinatorial optimization problems using parallel branch and bound, similar considerations regarding initial subdivisions occur, and ad hoc solutions are also common.

Let A be reduced to a standard region: $0 \le x_{:1} \le 1,\ 0 \le x_{:2} \le 2\sqrt{3}/3$. The original problem must be scaled to the standard feasible region. The latter is covered by the set of 28 triangles whose edge length is equal to 1/3 and whose vertexes $\chi_i,\ i = 1, \dots, 22$ have the coordinates $\chi_{i:1} \in \{j/6,\ j = -1, \dots, 7\}$, $\chi_{i:2} \in \{k \cdot \frac{\sqrt{3}}{6}, k = 1, 3,$ if j is odd and $k = 0, 2, 4,$ if j is even$\}$. The covering is shown in Figure 5.

The implementation of the version 1 is straightforward.

The implementation of version 2 includes solving the minimization problem (1). Let us consider the standard triangle defined by (2) and suppose that $f(\chi_3) \ge f(\chi_2) \ge f(\chi_1) = 0$. By sufficient scaling, the calculation of the bound value for each triangle may be reduced to the minimization on the standard triangle S:

$$b = \min_{x \in S} \max(-||x||,\ \psi_2 - ||x - \chi_2||,\ \psi_3 - ||x - \chi_3||)$$

where $\psi_i = f(\chi_i)/L,\ i = 2, 3.$

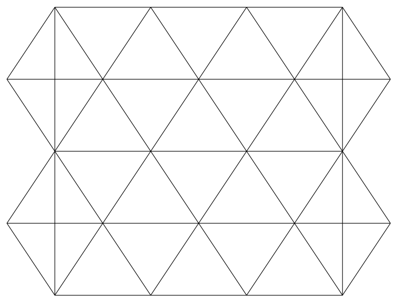

Figure 5: The initial cover in two dimensions.

Theorem 3 *For the lower bound b the following inequality holds:*

$$b \geq \beta_1 = (1 - \psi_2^2)/(2\psi_2 + \sqrt{3}).$$

Proof. Since b is monotone in ψ_2, ψ_3, we have:

$$b \geq \beta_1 = \min_{x \in S} \max(-||x||,\ \psi_2 - ||x - \chi_2||,\ \psi_2 - ||x - \chi_3||) =$$
$$\min_{h \geq 0} \min_{||x||=h} \max(-h,\ \psi_2 - ||x - \chi_2||,\ \psi_2 - ||x - \chi_3||).$$

Because of symmetry of the correct triangle, the condition $||x|| = h$ implies that the minimum of $\max(\psi_2 - ||x - \chi_2||,:\ \psi_2 - ||x - \chi_3||)$ is achieved on the symmetry axis. Therefore the solution to the minimization problem is:

$$\beta_1 = \min_{h \geq 0} \max(-h,:\psi_2 - \sqrt{1 + h^2 - h\sqrt{3}}) = (1 - \psi_2^2)/(2\psi_2 + \sqrt{3}),$$

and the minimum point is $x_1 = \sqrt{3}(1 - \psi_2^2)/(4\psi_2 + 2\sqrt{3}),\ x_2 = (1 - \psi_2^2)/(4\psi_2 + 2\sqrt{3})$.

Theorem 4 *For the lower bound b the following inequality holds:*

$$b \geq \beta_2 = \begin{cases} (\psi_3 - \sqrt{3}/2 + 1/(4(\psi_3 - \sqrt{3}/2)))/2, & \text{if } \psi_3 \leq (-1 + \sqrt{3})/4, \\ -(1 - \psi_3^2)/(1 + 2\psi_3), & \text{otherwise.} \end{cases}$$

Proof. Since b is monotone in ψ_2, ψ_3, we have the inequality

$$b \geq \beta_2 = \min_{x \in S} \max(-||x||,: -||x - \chi_2||,: \psi_3 - ||x - \chi_3||). \quad (3)$$

Because of symmetry with respect to $x_1 = 1/2$ the latter minimization problem may be reduced to one-dimensional minimization in the variable $0 \leq x_2 \leq \sqrt{3}/2$ with $x_{\cdot 1}$ fixed to 1/2 or in the variable $0 \leq x_1 \leq 1/2$ with $x_2$fixed to 0. In the first case a necessary condition for the minimum of (3)

$$\psi_3 - (\sqrt{3}/2 - x_2) = -\sqrt{1/4 + x_2^2}$$

is satisfied in the interior of the feasible region (at the point x where $(x_1 = 1/2,\ x_2 = -(\psi_3 - \sqrt{3}/2)/2 + 1/(8(\psi_3 - \sqrt{3}/2)))$, if $\psi_3 \leq (-1 + \sqrt{3})/4$. The minimal value is equal to $(\psi_3 - \sqrt{3}/2 + 1/(4(\psi_3 - \sqrt{3}/2)))/2$.

In case $\psi_3 > (-1+\sqrt{3})/4$ the equality $x_2 = 0$ is satisfied, and the necessary minimum condition is $-x_1 = \psi_3 - \sqrt{3/4 + (1/2 - x_1)^2}$ implying the bound $\beta_2 = -(1-\psi_3^2)/(1+2\psi_3)$ with the minimum point for (3) $x = ((1-\psi_3^2)/(1+2\psi_3),\ 0)$.

To implement the algorithm, bounds for each triangle may be obtained as rescaled values of $\max(\beta_1, \beta_2)$, where β_1 and β_2 are defined by the theorems above. The obtained bound may be sufficient to discard the current triangle. If not, the tighter bounds are calculated. The system of the following equations, expressing the necessary minimum conditions,

$$||x|| + \psi_3 - ||x - x_3|| = 0, ||x|| + \psi_2 - ||x - x_2|| = 0,$$

is numerically solved by the Newton method. For the initial point we use the better of the two points defined above. If the trajectory of the numerical solution crosses $x_2 = 0$ then the two separate equations obtained from the system by substituting x_2 with 0 should be solved and the better point chosen for the bound evaluation.

6 Results of experimental testing

In the recent review [6] the results of experimental testing of known algorithms are presented. The same test functions with the same estimates of Lipshitz constants and prescribed accuracies have been optimized by the algorithm proposed in present paper. We refer to [6] for a description of the conditions of the experiment. In the Table 1 the number of function evaluations for both versions of our algorithm is presented corresponding to the Table XI of ([6]). The results of the best algorithm according to [6] are included in the Table 1 for comparison. The second version of our algorithm (using the advanced bound calculation) is clearly superior.

The current implementation of the algorithm (in C) is experimental and has been developed to evaluate the method with repect to number of function evaluations, and the efficiency with respect the time of auxiliary computations has not been a mojor concern. Hence, do not tabulate the running times but merely note that these were less than a few seconds for all tests. The implementation is regarded as a step in the development of a parallel version of the algorithm, and in the parallel algorithm also the time eficiency will be addressed.

One remark should be made on the optimization of test function 4. In the table XI of [6] the value of requested accuracy is given 0.0141, while according to the conditions of the experiment it should be equal to 0.0283. Therefore it is not clear, with which value the calculations were performed in [6]. We have optimized this function with both values of requested accuracy and included in Table 1 version 4.1 (with 0.0141) and version 4.2 (with 0.0283).

Many tests used by the authors of [6] are close to the worst case conditions, e.g. the Lipshitz constant is very large, but it characterizes the behavior of the objective function in very small subregion near to the boarder, implying that the variation of function values in the remaining part of the feasible region with respect to the constant is insignificant. Of course, it is important to demonstrate the behavior of the algorithms under conditions similar to the worst case. However, in our opinion it is also important to present results with the functions which we consider to be typical representatives of difficult (but not

Test case [6]	Best of [6]	Version 1	Version 2
1	643	1006	489
2	167	217	37
3	3531	6622	2618
3.1	3953	7489	3245
3.2	3035	6160	2665
3.3	3689	7951	3387
4.1	*45	88	49
4.2	*45	103	41
5	73	112	53
6	969	3253	629
7	7969	12574	6370
8	301	613	255
9.	13953	22270	8759
9.1	14559	23119	9531
9.2	13281	21007	9002
9.3	12295	22207	8917
10	1123	2014	820
11	2677	4591	2222
12	12643	24997	10851
13	15695	28672	10643

Table 1: Number of function evaluations in optimizing the problems given in [6] using both versions of the proposed algorithm. "*" indicates that the prescribed accuracy in the experimental set-up of [6] is not clear.

Rastrigin			Two dim. Schubert		
ϵ	Version 1	Version 2	ϵ	Version 1	Version 2
0.392	1168	521	13.7	6895	2766
0.0392	2752	1407	1.37	18196	7776
0.00392	4180	2195	0.137	25180	11492
0.01	3934	2057	0.1	25435	11640

Table 2: Number of function evaluations of both versions of the algorithm for the Rastrigin and two dimensional Schubert functions with differing prescribed accuracies.

worst case) problems. Hence, we have performed experiments with two such functions, the Rastrigin test function and a two dimensional version of the Shubert function.

The Rastrigin test function is defined by

$$f(x) = \sum_{i=1}^{2} x_{\cdot i}^2 - cos(18x_{\cdot i}), \ -1 \leq x_{\cdot i} \leq 1, \ i = 1, 2.$$

It is widely used for testing of global optimization algorithms [12]the Shubert function [12], [6] is defined by :

$$f(x) = \sum_{i=1}^{2} \sum_{j=1}^{5} -j \cdot sin((j+1)x_{\cdot i} + j), \ -10 \leq x_{\cdot i} \leq 10, \ i = 1, 2.$$

Lipshitz constants of both functions were estimated using a quadratic mesh of 1000×1000 resulting in the values 27.7 and 96.8. The guaranteed accuracies of the passive algorithm defined by a quadratic mesh of 100×100 and the values of the Lipshitz constants above are 0.392 and 13.7. The numbers of function evaluations made by the algorithm with different levels of requested accuracy are presented in Table 2.

In all cases the value -2.0000 and point (0.0000, 0.0000) were found for the Rastrigin function. The value -24.0430 and point (-6.7708, 5.7813) were found for the second function in case of requested accuracy 13.7, and the value -24.0612 with point (-6.7773, 5.7910) were found in the other cases. The results show that the actual accuracy is much higher than the requested one. The number of function evaluations grows much slower than in worst case.

7 Conclusions

The Lipshitz model is one of best understood models of global optimization. Different ideas have been proposed and many algorithms have been implemented in the framework of this model. However, the available testing results show that the generalization from two to higher dimensions may be as complicated as the step from the implementation of one-dimensional algorithms to two dimensional algorithms. Hence, there is some pessimism regarding the perspective of the approach. In our view there is, however, also reasons for optimism. In our proposed version of the Branch and Bound Lipshitz technique some specific difficulties were overcome. The two dimensional version of the proposed technique compares favorably with other algorithms based on the Lipshitz model. A parallel version of the algorithm can immediately be implemented using the sequential C code. The multidimensional generalization seems promising, however, some experience with the two dimensional algorithm has to be accumulated and new possibilities, e.g. adaptive Lipshitz model [13], should be investigated prior to the extension to higher dimensions.

8 Acknowledgements

The authors want to thank an anonymous referee for a very detailed review with a number of valuable suggestions for clarifications and illustrations in the paper. The advice of the referee has improved the readability of the paper substantially.

References

[1] G.Wood B. Zhang and W.Baritompa. Multidimensional bisection: the performance and the context. *J. Global Optimization*, 3:337–358, 1993.

[2] W. Baritompa. Customizing method for global optimization-a geometric viewpoint. *J. Global Optimization*, 3:193–212, 1993.

[3] R. Brent. *Algorithms for minimization without derivatives.* Prentice-Hall, 1973.

[4] J. Clausen. Parallel branch and bound – principles and personal experience. In A. Migdalas et al., editor, *Parallel Computing in Optimization*, pages 239–267. KAP, 1997.

[5] E. Hansen. *Global Optimization Using Interval Analysis.* Marcel Dekker Inc., 1992.

[6] P. Hansen and B. Jaumard. Lipshitz optimization. In R.Horst and P.Pardalos, editors, *Handbook of Global Optimization*, pages 404–493. KAP, 1995.

[7] T. Hendriksen and K.Madsen. Parallel algorithms for global optimization. *Interval Computing*, 5:88–95, 1992.

[8] P. Pardalos Horst R. and N. Thoai. *Introduction to Global Optimization.* KAP, 1995.

[9] K. Holmqvist A. Migdalas and P. Pardalos. Parallelised heuristics for combinatorial search. In A. Migdalas et al., editor, *Parallel Computing in Optimization*, pages 269–294. KAP, 1997.

[10] C. Perttunen. A computational geometric approach to feasible region division in constrained global optimization. In *Proceedings of the 1991 IEEE Conference on Systems, Man, and Cybernetics.* IEEE, 1991.

[11] Horst R. and H. Tuy. *Global optimization - deterministic approaches, 2nd Edition.* Springer, 1992.

[12] A. Tőrn and A.Žilinskas. *Global optimization.* Springer, 1989.

[13] A. Žilinskas. Adaptive version of branch and bound type global optimization algorithm. In I.C.Parmee, editor, *Poster Proceedings of the Third International Conference Adaptive Computing in Design and Manufacture*, pages 34–37. University of Plymouth, 1998.

[14] G. Wood. Multidimensional bisection applied to global optimization. *Computers Math. Appl.*, 21:161–172, 1991.

[15] G. Wood. The bisection method in higher dimensions. *Mathematical Programming*, 55:319–337, 1992.