Numerical Algorithms **37:** 391–399, 2004.

A multidimensional branch-and-prune method for interval global optimization *

Tamás Vinkó [a] and Dietmar Ratz [b]

[a] *Research Group on Artificial Intelligence of the Hungarian Academy of Sciences, and University of Szeged, H-6720 Szeged, Aradi vértanúk tere 1, Hungary*
E-mail: tvinko@inf.u-szeged.hu

[b] *Department of Applied Informatics and Formal Descriptions, University of Karlsruhe, Karlsruhe, Germany*
E-mail: ratz@aifb.uni-karlsruhe.de

In this paper a new multidimensional extension of the recently developed one-dimensional enclosure method called kite is given for interval global optimization. A more sophisticated version of the pruning technique based on the kite method is introduced. By the new componentwise approach all the one-dimensional theoretical results and procedures can be used in the higher-dimensional case. The possibilities in the implementation of the new algorithm together with numerical results on 40 standard test problems are presented.

Keywords: interval methods, global optimization, inclusion function, pruning, kite

AMS subject classification: 90C30, 65K05

1. Introduction

In a recent paper a new inclusion function called *kite* enclosure has been developed and investigated for global optimization for the one-dimensional case [12]. The aim of the present paper is to give an extension of those results for the multidimensional problems. Consider the problem of finding all solutions x^* of

$$\min_{x \in X} f(x), \tag{1}$$

where the objective function $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and $X \subseteq D$ is the search box representing bound constraints for x. We wish to find the set of all global minimizers x^* and the global minimum value $f(x^*)$. With interval global optimization algorithms based on branch-and-bound methods [3,4,9] we can provide guaranteed and reliable solutions for the problem (1).

In the following real numbers are denoted by lower case letters $(a, b, \ldots)$ and real bounded and closed n-dimensional intervals by capital letters $(X, Y, \ldots)$. The set of compact intervals is denoted by $\mathbb{I} := \{[a, b] \mid a \leqslant b;\ a, b \in \mathbb{R}\}$. In general, the lower and the upper bounds of an interval $X = X_1 \times \cdots \times X_n$ are denoted by $\underline{X}$ and $\overline{X}$, re-

* This work has been supported by the grant OTKA T 034350.

spectively. The range of the function f on X is denoted by $f(X)$. We call a function $F:\mathbb{I}^n \to \mathbb{I}$ an *inclusion function* of f in X if $x \in Y$ implies $f(x) \in F(Y)$ for all $Y \in \mathbb{I}(D) \subseteq \mathbb{I}^n$, where $\mathbb{I}(D)$ stands for all intervals in D. Applying automatic differentiation [2,4,6] we are able to compute inclusion functions for the derivatives without additional human interaction. An enclosure of the gradient vector $f'(y)$ is denoted by $F'(Y)$, while the ith component of this vector by $F_i'(Y) = [L_i, U_i]$. For all $i = 1, \ldots, n$ the inequality $L_i U_i < 0$ is supposed in this work.

The detailed forms of the branch and bound algorithm used in this paper can be found in [2–4,9]. Here a new accelerating tool is investigated which is based on the pruning possibilities of the one-dimensional kite method.

2. Componentwise extension of the kite enclosure method

In the one-dimensional case the kite inclusion function is simply the simultaneous usage of the linear boundary value form [7] and the centered form [8] with a proper selection of the center point. For higher dimension the extension of the linear boundary value form exists (see [5] for details). The centered form is also available for multidimensional functions. However, the simultaneous usage of them is in general difficult to realize. In the following an effective and easier to implement way of the extension is given.

In [11] the centered form and a pruning method based on slopes is developed for the multidimensional case in a componentwise manner. The extension of the kite enclosure together with the pruning effect is also possible in this way. In the following this approach is discussed.

Let $f: D \subseteq \mathbb{R}^n \to \mathbb{R}$ and $Y = Y_1 \times \cdots \times Y_n \subseteq D$. Let us define $g_i: Y_i \subseteq \mathbb{R} \to \mathbb{I}$ ($i \in \{1, \ldots, n\}$) by

$$g_i(w) := f(Y_1, \ldots, Y_{i-1}, w, Y_{i+1}, \ldots, Y_n), \quad w \in Y_i.$$

Using this one-dimensional interval valued function we are able to use the one-dimensional version of the kite enclosure method. If we have $V \supseteq g_i(\underline{Y}_i)$, $W \supseteq g_i(\overline{Y_i})$ and $Z \supseteq g_i(c_i)$ where $(c_i \in Y_i)$, then the componentwise kite enclosure can be built up with the usage of the componentwise centered form, i.e.

$$\underline{F}_{\mathrm{CF}}(Y, c, i) = \underline{Z} + F_i'(Y)(Y_i - c_i), \quad c_i \in Y_i, \tag{2}$$

and the componentwise linear boundary value form

$$\underline{F}_{\mathrm{LBVF}}(Y, i) = \frac{U_i \underline{V} - L_i \underline{W}}{U_i - L_i} + \left(\overline{Y_i} - \underline{Y}_i\right) \frac{L_i U_i}{U_i - L_i}. \tag{3}$$

The simultaneous usage of the formulae (2) and (3) leads to the following result.

Theorem 1. Let $\underline{F}_K(Y, c, i) = \min\{y_R(c, i), y_T(c, i)\}$, where $c \in Y$,

$$y_R(c, i) = \frac{U_i \underline{V} - L_i \underline{Z} + U_i L_i (c_i - \underline{Y}_i)}{U_i - L_i},$$

$$y_T(c,i) = \frac{U_i \underline{Z} - L_i \underline{W} + U_i L_i (\overline{Y_i} - c_i)}{U_i - L_i},$$

$Z \supseteq g_i(c_i)$, $V \supseteq g_i(\underline{Y}_i)$ and $W \supseteq g_i(\overline{Y_i})$, and $i = 1, \ldots, n$. Then

$$\max\left\{\underline{F}_{\mathrm{LBVF}}(Y,i), \underline{F}_{\mathrm{CF}}(Y,c,i)\right\} \leqslant \underline{F}_K(Y,c,i) \leqslant \underline{f}(Y) \tag{4}$$

hold for every $i = 1, \ldots, n$.

Proof. One can proceed the proof of proposition 2 from [12] for all $i = 1, \ldots, n$. □

Note that theorem 1 tells that the componentwise kite method is not worse than the componentwise centered form or the componentwise linear boundary value form (for the same component Y_i of the interval Y).

In formula (4) the parameter c can be set to be optimal. That is we can find the point c^* such that

$$\underline{F}_K(Y,c^*,i) = \max_{c \in Y} \underline{F}_K(Y,c,i) = \max_{c \in Y} \min\left\{y_R(c,i), y_T(c,i)\right\}. \tag{5}$$

To obtain this optimal c^* theorem 1 of [12] can be used for every coordinate direction. Note that here the value c^* depends on the interval Y and the direction i, i.e. if $i \neq j$ then the value $c^*(Y,i)$ is usually not equal to the value $c^*(Y,j)$. From [12] we know that the point $c^*(Y,i)$ is not always unique and it can be given as a solution of a nonlinear equation, which has the form $\alpha_i g_i(z) + \beta_i = 0 \quad (\alpha_i, \beta_i, z \in \mathbb{R})$. In the implementation we usually do not compute the exact value of $c^*(Y,i)$, only an approximation of it is given using the same technique as in the one-dimensional case. For details see [12].

From the formula (4) or even from (5) a lower bound for $f(X)$ can be obtained: the value $\max_{1 \leqslant i \leqslant n} \underline{F}_K(Y,c,i)$ is always less than or equal to $\underline{f}(Y)$. However, the centered form or the multidimensional extension of the linear boundary value form can give a better (greater) lower bound for the range of the objective function in general. To compute a lower bound for the value $f(Y)$ with the componentwise kite inclusion $3n$ function evaluations (2 evaluations at the endpoints and 1 evaluation at the centers) and one gradient evaluation are needed. Thus, the componentwise versions of the inclusion functions above are not recommended to be used in the interval global optimization algorithms. It is why this approach is used better in the construction of a new accelerating tool.

3. Componentwise pruning in higher dimension

Using the computed value needed for the construction of the componentwise kite inclusion function a pruning method can be developed. The following theorem proves the correctness of the necessary formulae.

Theorem 2. Let $Y \subseteq X \subseteq \mathbb{I}^n$ be the current considered subinterval, $c \in Y$, $F'(Y)$ be an enclosure of the gradient of $f(Y)$, and $\tilde{f}$ be the current guaranteed upper bound for the global minimum value. If we have $Z \supseteq g_i(c_i)$, $V \supseteq g_i(\underline{Y}_i)$ and $W \supseteq g_i(\overline{Y_i})$,

$$p_i = \underline{Y}_i + \frac{\tilde{f} - \underline{V}}{L_i}, \qquad q_i = c_i + \frac{\tilde{f} - \underline{Z}}{U_i},$$
$$r_i = c_i + \frac{\tilde{f} - \underline{Z}}{L_i}, \qquad s_i = \overline{Y_i} + \frac{\tilde{f} - \underline{W}}{U_i},$$

then for every $i \in \{1, \ldots, n\}$ the following statements hold:

(a) If $\tilde{f} < \min\{\underline{V}, \underline{W}, \underline{Z}\}$ then all the global minimizer points of Y are contained in the intervals $Y_1 \times \cdots \times Y_{i-1} \times [p_i, q_i] \times Y_{i+1} \times \cdots \times Y_n$ and $Y_1 \times \cdots \times Y_{i-1} \times [r_i, s_i] \times Y_{i+1} \times \cdots \times Y_n$.

(b) If $\underline{W} \leqslant \tilde{f} < \min\{\underline{V}, \underline{Z}\}$ then all the global minimizer points of Y are contained in the intervals $Y_1 \times \cdots \times Y_{i-1} \times [p_i, q_i] \times Y_{i+1} \times \cdots \times Y_n$ and $Y_1 \times \cdots \times Y_{i-1} \times [r_i, \overline{Y_i}] \times Y_{i+1} \times \cdots \times Y_n$.

(c) If $\underline{V} \leqslant \tilde{f} < \min\{\underline{Z}, \underline{W}\}$ then all the global minimizer points of Y are contained in the intervals $Y_1 \times \cdots \times Y_{i-1} \times [\underline{Y}_i, q_i] \times Y_{i+1} \times \cdots \times Y_n$ and $Y_1 \times \cdots \times Y_{i-1} \times [r_i, s_i] \times Y_{i+1} \times \cdots \times Y_n$.

(d) If $\underline{Z} \leqslant \tilde{f} < \min\{\underline{V}, \underline{W}\}$ then all the global minimizer points of Y are contained in the interval $Y_1 \times \cdots \times Y_{i-1} \times [p_i, s_i] \times Y_{i+1} \times \cdots \times Y_n$.

(e) If $\max\{\underline{W}, \underline{Z}\} \leqslant \tilde{f} < \underline{V}$ then all the global minimizer points of Y are contained in the interval $Y_1 \times \cdots \times Y_{i-1} \times [p_i, \overline{Y_i}] \times Y_{i+1} \times \cdots \times Y_n$.

(f) If $\max\{\underline{V}, \underline{Z}\} \leqslant \tilde{f} < \underline{W}$ then all the global minimizer points of Y are contained in the intervals $Y_1 \times \cdots \times Y_{i-1} \times [\underline{Y}_i, s_i] \times Y_{i+1} \times \cdots \times Y_n$.

(g) If $\max\{\underline{V}, \underline{W}\} \leqslant \tilde{f} < \underline{Z}$ then all the global minimizer points of Y are contained in the intervals $Y_1 \times \cdots \times Y_{i-1} \times [\underline{Y}_i, q_i] \times Y_{i+1} \times \cdots \times Y_n$ and $Y_1 \times \cdots \times Y_{i-1} \times [r_i, \overline{Y_i}] \times Y_{i+1} \times \cdots \times Y_n$.

Proof. (a) Let $x^* \in Y \subseteq X$ be the global minimizer point and let $i \in \{1, \ldots, n\}$ be arbitrary but fixed. First, we have to show that $p_i \leqslant x_i^*$ holds. Since $\tilde{f} < \underline{V}$ is supposed and $\underline{V} \leqslant f(x^*)$ then $x_i^* \neq \underline{Y}_i$. From the inequalities

$$L_i \leqslant \frac{f(x^*) - \underline{V}}{x_i^* - \underline{Y}_i} \leqslant \frac{\tilde{f} - \underline{V}}{x_i^* - \underline{Y}_i}$$

we have $(x_i^* - \underline{Y}_i)L_i \leqslant \tilde{f} - \underline{V}$. Then since

$$x^* \geqslant \frac{\tilde{f} - \underline{V}}{L_i} + \underline{Y}_i = p_i$$

holds because $L_i < 0$ was supposed.

To prove that x_i^* is not included in the open interval (q_i, r_i) first we suppose that $x_i^* < c_i$ holds. Then from

$$U_i \geqslant \frac{f(x^*) - \underline{Z}}{x_i^* - c_i}$$

we have $U_i(x_i^* - c_i) \leqslant f(x^*) - \underline{z} \leqslant \tilde{f} - \underline{Z}$. From these inequality

$$x_i^* \leqslant c_i + \frac{\tilde{f} - \underline{Z}}{U_i} = q_i$$

holds. Now suppose that $x_i^* > c_i$. Then from

$$L_i \leqslant \frac{f(x^*) - \underline{Z}}{x_i^* - c_i} \leqslant \frac{\tilde{f} - \underline{Z}}{x_i^* - c_i}$$

we have $(x_i^* - c_i)L_i \leqslant \tilde{f} - \underline{Z}$. Then

$$x_i^* \geqslant c_i + \frac{\tilde{f} - \underline{Z}}{L_i}$$

holds because $x_i^* - c_i > 0$ and $L_i < 0$ were supposed. The case $x_i^* = c_i$ is not possible, because $\underline{Z} = \underline{g_i}(c_i) > \tilde{f}$ was supposed and $\tilde{f} \geqslant f(x^*)$.

Finally, from

$$U_i \geqslant \frac{f(x^*) - \underline{W}}{x_i^* - \overline{Y_i}} \geqslant \frac{\tilde{f} - \underline{W}}{x_i^* - \overline{Y_i}}$$

we have $(x_i^* - \overline{Y_i})U_i \leqslant \tilde{f} - \underline{W}$. Then

$$x_i^* \leqslant \overline{Y_i} + \frac{\tilde{f} - \underline{W}}{U_i}$$

holds because $U_i > 0$ by our earlier assumption. With this we completed the proof of the case (a).

The proof of cases (b)–(g) are similar to the case (a) thus we have omitted them. □

Based on the theorem 2 we can conclude that in any case if $Y_i = [\underline{Y}_i, \overline{Y_i}]$ and $((p_i > \underline{Y}_i) \wedge (s_i < \underline{Y}_i))$ or $((q_i < \underline{Y}_i) \wedge (r_i > \overline{Y_i}))$ then the whole subinterval Y can be rejected: it does not contain any global minimizer points.

4. Proposed algorithm

Now we give the algorithmic description of the new branch-and-prune global optimization method based on the above results.

Step A. Let X be the starting interval. Compute $F(X)$, $F'(X)$, initialize *WorkList* $= \{(X, \underline{F}(X), F'(X))\}$, *ResultList* $= \{\}$, and the guaranteed upper bound $\tilde{f} = F(c)$ for the global minimum value.

Step B. While *WorkList* is not empty do the following steps.

Step C. Get $(Y, \underline{F}(Y), F'(Y))$ from the *WorkList* and for every coordinate direction of Y do the following steps.

Step C.1. Compute the componentwise kite enclosure for the ith coordinate.

Step C.2. Apply the pruning method based on the kite for the ith coordinate.

Step D. For the subintervals U_i $(i = 1, \ldots, m \leqslant n + 1)$ produced by the pruning do the following steps.

Step D.1. Compute $F(U_i)$, $F'(U_i)$. Apply the monotonicity and the midpoint tests.

Step D.2. Compute the centered form for U_i (and improve $\tilde{f}$ if it is possible).

Step D.3. If the stopping criterion holds for the current interval then put it to the *ResultList* else put it (together with the values $\underline{F}(U_i)$, $F'(U_i)$) to the *WorkList*.

Step E. Go back to step B.

First of all it must be emphasized that in this new algorithm the componentwise kite enclosure is used as a pruning step (i.e. an accelerator) and not only as an inclusion function. To obtain a (possibly) better enclosure of the objective function on the current subinterval the original centered form is used (in step D.2). This is done so since the centered form gives usually a better (higher) lower bound than the componentwise kite method. However, in the step C the information needed for the kite enclosure could be used to update the value $\tilde{f}$. Also, the candidate subinterval Y can be rejected if the inequality $\tilde{f} < \underline{F}_K(Y, c, i)$ (i.e. a range test) is fulfilled.

In the step C.2 a special splitting technique is used which was introduced in [10]. This performs one component step according to the following scheme:

1. Let $V, W \subseteq Y_i$ the intervals produced by the pruning step.
2. If $W = V = \emptyset$, then stop (there is no solution in Y).
3. If $V \neq \emptyset$, then set $Y_i := V$ and store Y.
4. Set $Y_i := W$ and continue with next i.

With this method the pruning procedure produces at most $n + 1$ subintervals. If pruning is possible in an iteration step, then the inclusion of the derivative is not computed in the next iteration (which deals with the previously shrunk interval). Note that the step C performs a bisection if no pruning is possible.

In step C it is not necessary to compute Y_i in the fixed order $i = 1, \ldots, n$. One can use a sorted index vector to obtain different orders of the components. In our experience

we have tried to use sorted index vectors generated by interval branching rules from A to D [1]. We have found that the best choice is the rule C based on the maximization of the merit function

$$D(i) = \mathrm{w}\big(F_i'(Y)\big(Y_i - \mathrm{mid}(Y_i)\big)\big), \tag{6}$$

where $\mathrm{w}(X)$ and $\mathrm{mid}(X)$ denote the width and the midpoint of the interval X, respectively. The new index vector $t = (t_1, \ldots, t_n)$ with $t_k \in \{1, \ldots, n\}$ and $t_i \neq t_j$ for $i \neq j$, satisfies $D(t_k) \geqslant D(t_{k+1}), k = 1, \ldots, n-1$.

According to our considerations above we can state that using the proposed algorithm we cannot loose global minimizer points which are in the starting interval X. In the pruning step for a given interval Y if the value m is equal to zero then there is no global (with respect to X) minimizer of f in Y.

5. Numerical results

This section deals with the discussion of the numerical results based on the componentwise realization of the multidimensional kite enclosure method and its pruning technique. The goal of the test is to demonstrate the effect and the behavior of the new accelerating tool compared to the traditional algorithm. The implementation of the algorithm given in the section 4 has been done on a Pentium III, 1 GHz computer under Linux operating system and in the environment of the C++ Toolbox for Verified Computing [2]. For the comparison the new algorithm was used without the step C and in step D the value m was set to be 2 (i.e. only bisection was used). In the computational experiments 40 standard test functions were taken from the literature. The stopping criterion was fulfilled when the maximal width of the current subinterval was smaller than 10^{-6} (except for the problems GP, Sch27, Sch214, G7, R5, R6, R7, R8, EX2 where this number was 10^{-2}.)

Our numerical experience shows that in the computation of the componentwise kite enclosure using formula (5) the resulted intervals $g_i(\underline{Y}_i)$, $g_i(c_i)$, $g_i(\overline{Y_i})$ could be very large. In this case the pruning usually cannot be used because of that large overestimations. For this reason the algorithm leaves the pruning step (step C) and do a bisection if one of the width of the intervals $g_i(\underline{Y}_i)$, $g_i(c_i)$, $g_i(\overline{Y_i})$ is greater than a specified heuristic parameter. In our tests this parameter was set to be $\max\{D(t_1), 100\}$, where D is the merit function defined in (6). Using this modification the total computational effort can be reduced.

Both algorithms were able to solve all the test problems. The numerical results are summarized in table 1. The given efficiency indicators are the number of the function evaluations (F-eval), the number of the derivative evaluations (D-eval), the maximal list length necessary (MLL) and the CPU time in seconds (CPUt). In the last but one row $\sum$ denotes the sum of the given efficiency indicators followed by the relative compound indicator for the new method compared to the old one as per cents. The last row contains the average of the percentages (AoP) for the respective columns.

Table 1
Numerical results of the basic and the kite pruning algorithm.

Problem		F-eval.			D-eval.			MLL			CPUt		
Name	dim.	old	new	%	old	new	%	old	new	%	old	new	%
S5	4	281	450	160	179	177	98	9	9	100	0.36	0.55	152
S7	4	291	478	164	183	184	100	12	12	100	0.50	0.79	158
S10	4	291	478	164	183	184	100	12	12	100	0.71	1.12	157
H3	3	1,338	1,232	92	889	683	76	21	13	61	1.07	0.99	92
H6	6	3,654	4,705	128	2,399	1,767	73	118	79	66	9.01	10.87	120
GP	2	15,991	24,043	150	8,653	7,696	88	798	761	95	8.32	10.76	129
SHCB	2	1366	1,896	138	859	750	87	60	57	95	0.31	0.38	122
THCB	2	874	848	97	563	291	51	24	17	70	0.15	0.13	86
BR	2	1,278	769	60	831	297	35	17	10	58	0.26	0.14	53
RB	2	460	559	121	283	252	89	11	12	109	0.07	0.07	100
RB5	5	2,582	2,775	107	1,601	1,445	90	58	58	100	1.93	1.93	100
L3	2	2,522	2,481	98	1,629	671	41	119	98	82	1.40	1.22	87
L5	2	587	933	158	385	285	74	29	37	127	0.36	0.53	147
L8	3	237	282	118	153	150	98	9	9	100	0.13	0.15	115
L9	4	315	369	117	203	200	98	12	12	100	0.27	0.30	111
L10	5	393	453	115	253	251	99	15	15	100	0.48	0.54	112
L11	8	627	709	113	403	401	99	24	24	100	1.90	2.09	109
L12	10	783	878	112	503	501	99	30	30	100	3.71	4.15	111
L13	2	162	229	141	103	96	93	7	6	85	0.04	0.05	125
L14	3	243	329	135	153	145	94	10	11	110	0.10	0.13	130
L15	4	323	436	134	203	194	95	13	14	107	0.21	0.27	128
L16	5	388	514	132	243	238	97	16	12	75	0.37	0.46	124
L18	7	542	708	130	339	334	98	22	16	72	0.91	1.13	124
Sch21	2	2,004	2,125	106	1,249	868	69	36	31	86	0.47	0.45	95
Sch31	3	253	357	141	153	159	103	3	5	166	0.08	0.11	137
Sch25	2	649	736	113	415	323	77	8	8	100	0.10	0.10	100
Sch27	3	708,262	28,505	4	472,269	15,726	3	45,364	1,901	4	9,597.84	53.44	0
Sch214	4	15,771	11,692	74	10,317	6,399	62	382	355	92	6.92	4.47	64
Sch218	2	2,022	2,393	118	1,215	1,140	93	18	18	100	0.26	0.28	107
Sch32	3	866	863	99	545	411	75	13	10	76	0.21	0.20	95
Sch37	5	8,830	8,766	99	5,887	5,823	98	32	32	100	5.84	5.67	97
Sch37	10	559,102	557,054	99	372,735	370,687	99	1,024	1,024	100	2,070.49	2,026.59	97
G5	5	14,590	1,741	11	9,727	705	7	32	32	100	11.28	1.25	11
G7	7	43,774	11,578	26	29,183	2,855	9	128	128	100	64.03	13.45	21
R4	2	2,454	1,390	56	1,615	633	39	72	32	44	0.41	0.20	48
R5	3	33,386	14,727	44	22,251	8,893	39	1,024	512	50	28.48	10.16	35
R6	5	52,558	31,543	60	35,023	19,881	56	1,024	768	75	91.02	48.48	53
R7	7	71,730	44,337	61	47,795	28,349	59	1,024	768	75	203.42	112.16	55
R8	9	90,902	79,971	87	60,567	51,541	85	1,024	896	87	409.26	328.78	80
EX2	5	425,349	690,379	162	279,673	213,027	76	13,236	12,007	90	2,428.82	2,556.78	105
$\sum$		2,068,030	1,534,711	74	1,371,812	744,612	54	65,890	19,851	30	14,951.50	5,270.99	35
AoP				106			76			89			97

Summarizing the results we can conclude that the number of function evaluations were larger for 24 test functions for the new method. It is understandable since the construction of the componentwise kite needs more function evaluations at the borders of the considered subintervals. In such cases the new method cannot solve the problems faster than the old one.

The number of the derivative evaluations were smaller for the new method in almost every cases. Actually, this number refers to the number of the iteration steps in the algorithms, thus we can see that the new and the old method converge in a different path in the branch-and-bound tree to the global optimizier points.

The memory complexity was smaller for the new method, it puts much less subintervals to the list of the candidate intervals. The total CPU time used by the new algorithm is 35% of that of the old method showing a computation time saving for the whole set of test problems. However, calculating the average of the percentages for each of the problems we obtain 3% improvement. From these two indicators we can conclude that the new algorithm can perform better on the harder to solve problems.

We can conclude from the indicators that the new method is not better for the Shekel functions (S5, S7, S10). On the other hand, for the problems of Ratz (R4–R8) the proposed algorithm performs well. The largest improvements were achieved for the Schwefel-27 (Sch27) and the Griewank problems (G5, G7).

Summarizing the numerical results we can conclude that the new algorithm using the pruning technique proved to be better than the traditional one on the test problems. The improvements are larger on the harder to solve problems.

References

[1] T. Csendes and D. Ratz, Subdivision direction selection in interval methods for global optimization, SIAM J. Numer. Anal. 34(3) (1997) 922–938.

[2] R. Hammer, M. Hocks, U. Kulisch and D. Ratz, *C++ Toolbox for Verified Computing I: Basic Numerical Problems: Theory, Algorithms, and Programs* (Springer, Berlin, 1995).

[3] E. Hansen, *Global Optimization Using Interval Analysis* (Marcel Decker, New York, 1992).

[4] R.B. Kearfott, *Rigorous Global Search: Continuous Problems* (Kluwer, Boston, 1996).

[5] F. Messine and J.-L. Lagouanelle, Enclosure methods for multivariate differentiable functions and application to global optimization, J. Universal Computer Sci. 4(6) (1998) 589–603.

[6] R.E. Moore, *Methods and Applications of Interval Analysis* (SIAM, Philadelphia, PA, 1979).

[7] A. Neumaier, *Interval Methods for Systems of Equations* (Cambridge Univ. Press, Cambridge, 1990).

[8] H. Ratschek and J. Rokne, *Computer Methods for the Range of Functions* (Ellis Horwood, Chichester, 1984).

[9] H. Ratschek and J. Rokne, *New Computer Methods for Global Optimization* (Ellis Horwood, Chichester, 1988).

[10] D. Ratz, Automatische Ergebnisverifikation bei globalen Optimierungsproblemen, Ph.D. thesis, Universitaet Karlsruhe (1992).

[11] D. Ratz, *Automatic Slope Computation and its Application in Nonsmooth Global Optimization* (Shaker-Verlag, Aachen, 1998).

[12] T. Vinkó, J.-L. Lagouanelle and T. Csendes, A new inclusion function for optimization: Kite – the one-dimensional case, J. Global Optimization (2004) in press.