ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2005, Vol. 34, No. 2

ON EFFICIENCY OF ADAPTIVE SEARCH IN OPTIMIZATION OVER SMALL IMPLICITLY DEFINED FEASIBLE REGION

Antanas Žilinskas[1,2], Aušra Mackutė[2]

[1]*Institute of Mathematics and Informatics*
Akademijos str. 4, LT-08663 Vilnius, Lithuania

[2]*Vytautas Magnus University*
Vileikos str. 8, LT-44404 Kaunas, Lithuania

Abstract. An optimization problem related to optimal design of processes in oil industry is considered. The problem is difficult because of small, implicitly defined feasible region. In such a case it is difficult not only to construct a rational algorithm for search for minimum, but also to construct an algorithm for search for feasible points. Several algorithms for finding feasible points are proposed as well as several algorithms for optimization in a region approximated using points scattered in the region. A set of test functions is constructed to model the considered industrial optimization problems which normally are not suitable for testing of algorithms because of computational intensity. Testing results are presented, and conclusions about algorithms efficiency are drawn.

1. Indroduction

A standard optimization problem consists of feasible region defined by equalities and inequalities, and of objective function defined not only inside but also outside of the feasible region. However, in some applications there occur optimization problems with implicitly defined feasible regions and objective functions not defined outside of the feasible region.

We need to find the minimum value and a minimizer for the problem

$$\min_{X \in A} f(X), \qquad (1)$$

where $f(X)$ is not defined for $X \notin A$, $A \subset B \subset R^n$, $B = \{X : b_i^- \le x_i \le b_i^+\}$, and A itself is defined by an indicator function $I(X)$

$$I(X) = \begin{cases} 1, X \in A, \\ 0, X \notin A. \end{cases}$$

Important properties of the optimization problem are discussed in [3], [4]. The aim of this paper is to develop algorithms suitable for a class of problems similar to that presented in [3]. For such research project a class of test problems satisfying assumptions below is needed.

The problems of black box optimization are typical for process engineering in case the physical and economical properties are modelled by software packages allowing limited access to the implemented models. For example, the only output of the package is either the objective function value or indication that the input variables are infeasible. An optimization problem of such a type, related to industrial processing raw hydrocarbon feed stock into oil and gas products, is considered in [3]. The design and optimization of such a process is difficult due to a combination of features of the process and the models used, both for modelling the physical behaviour of the process and for deriving cost estimates of a given configuration and set of operating conditions [3]. The objective function value is calculated using the modelling package Jacaranda [1]. The objective is the profit of the process, which is calculated using the parameters obtained via modelling the physical processes, and market data. The package returns the objective function value (profit with sign minus) for the design parameters granting requested quality of the products. The package returns 10^{20} in the case either the process is physically impossible or the products quality is not satisfactory. The reasonable intervals of the variables are known, and they constitute the hyper-rectangle set B. However, a large part of B is not feasible since the parameter combinations are not compatible with physical feasibility of desired processes. Even larger part of B is not feasible because of not acceptable products quality.

To develop efficient algorithms for problems similar to the problem in [1], [3], a class of test functions modelling properties of the considered problem is needed because experimentation with the original problem is prohibitive computationally intensive. Most important property of the original problem is

size of the feasible region A. It is so small with respect to B that even finding feasible points is challenging. However, the problem is not hopeless since it is possible to construct enclosures for A. A $\theta-$enclosure of A, denoted by A_θ, is defined by the inclusions $A \subset A_\theta \subset B$, $0 \le \theta \le 1$; $A_\theta \subset A_\pi$, $\pi < \theta$; $A \equiv A_1$. The hypervolume of A_0 constitutes about one percent of the hypervolume of B. The sizes of the enclosures for different θ of the problem of [3] are shown in Figure 1.

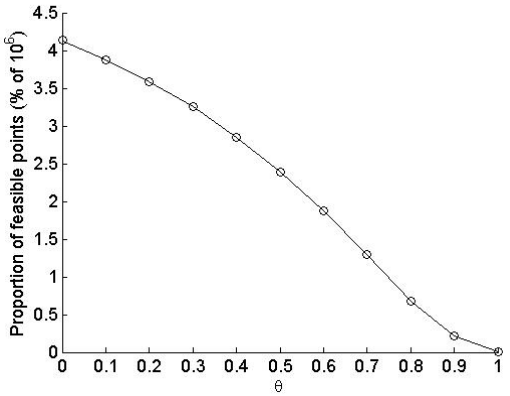

Figure 1. Hypervolume of enclosure depending on θ [3]

Discontinuity and other irregularities of the objective function of the original problem are caused by numerical methods used in modelling of physical processes. Therefore the construction of a descent trajectory is difficult even from a feasible starting point. For example, Figure 2 illustrates not only the discontinuity of objective function but also the presence of local minima close to the points of discontinuities. Because of these properties of the objective function, the application of gradient based descent methods does not seem possible. However, these difficulties are not fatal for direct search methods.

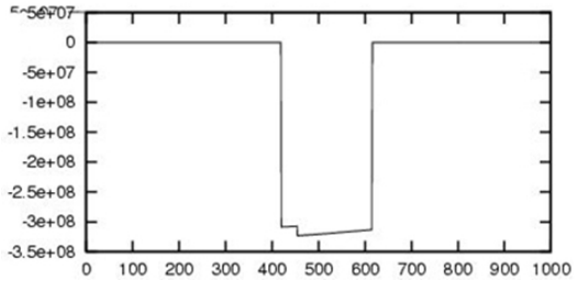

Figure 2. Cut of the objective function in x_2

The feasible region A of the original problem in [3], is not disjoint but with a non-smooth boarder. The conclusion that the set A_1 is not disjoint is corroborated by the projection of points scattered in A_1 to the plane defined by two first principal coordinates; see Figure 3.

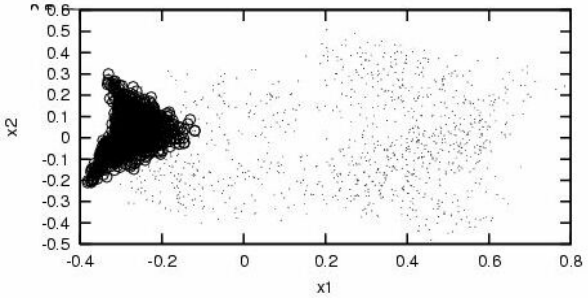

Figure 3. Projection of trial points to the plane of two first principal components. Feasible points are indicated with large circles, and infeasible points with small dots

2. Test problems

To assess efficiency of optimization algorithms experimentally, test problems similar to the original industrial problem are needed. We take into account the following properties of the original problem. The feasible region should be small, e.g. with ratio $vol(A)/vol(B)$ of order $10^{-4} - 10^{-6}$, where vol(.) denotes hypervolume. According to the discussion in the previous section the enclosures of the feasible region should be available. To simplify comparison of algorithms, we construct test functions with known value of global minimum and known global minimizer.

Two test problems below represent the minimization problems with small implicitly defined feasible regions whose enclosures can be controlled via parameter θ. The test functions differ from the real world objective function discussed above with respect to the smoothness. However, this difference is not essential since we consider search algorithms not using gradients/smoothness of the objective functions.

For the first test problem (Problem I) we use well known Rosenbrock function but with small nonconvex feasible region

$$f(X) = 100\left(x_2 - x_1^2\right)^2 + \left(x_1 - 1\right)^2, \qquad (2)$$

$$A_\theta = \left\{ \begin{array}{l} X : X \in B \subset R^2, \\ 0.01\left(x_1 - 1\right)^2 + 100\left(x_2 - x_1^2\right)^2 \le 1 - 0.9995\theta \end{array} \right\}$$

where $A = A_1$, $B = \left\{X : -2 \le x_i \le 2, \ \ i = 1,2\right\}$. Figure 4 illustrates the feasible region and its enclosure with $\theta = 0$.

The minimum value is equal to 0 and it is achieved at one point X=(1, 1). For B in (2) re-scaled to a unit cube, the minimum point in new scales is (0.75, 0.75). The largest enclosure A_θ constitutes approximately 3.5% of B while the estimate of the ratio is $vol(A)/vol(B) = 0.00019$; it is obtained from 10^6 random trials with uniform distribution over B. The sizes of enclosures for different θ are shown as dot line (3) in Figure 5.

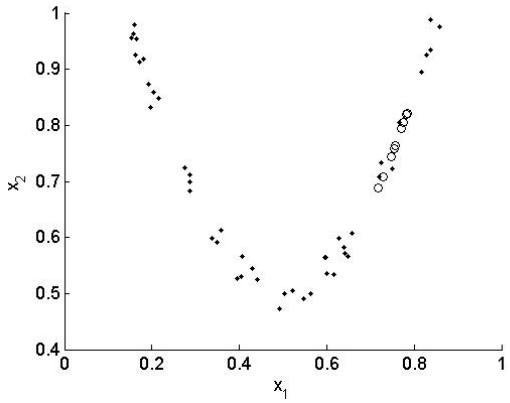

Figure 4. Feasible region of problem (2); (.) – represent points in enclosure with $\theta = 0$ and (o) – represent points in enclosure with $\theta = 1$

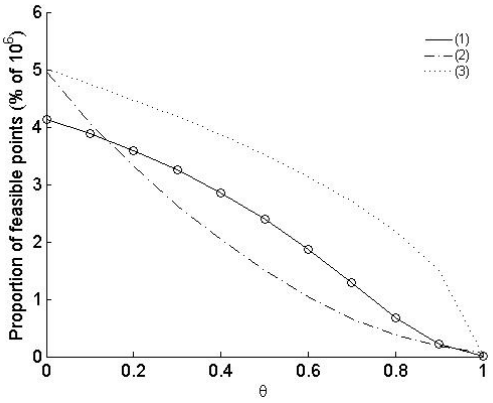

Figure 5. Hypervolume of enclosure depending on θ; (1) – Industrial problem, (2) – Problem II, (3) – Problem I

To test the algorithms in a case of higher dimensionality, the second test problem is considered. The objective function of this test problem (Problem II), the feasible region, and its enclosures are defined below:

$$f(X) = -100(x_1 + 1)^2 + 121\sum_{2}^{5}(x_i - 1)^2 + 0.2086, \qquad (3)$$

$$A_\theta = \left\{ \begin{matrix} X : X \in B \subset R^2, \\ 5|x_1 + x_2| + |x_1 - x_2 + 2| \le 2 - 1.8\theta \end{matrix} \right\},$$

where $A = A_1$ and $B = \{X : -2 \le x_i \le 2, i = 1,...,5\}$. Figure 6 illustrates convexity of the feasible region and its enclosure with $\theta = 0$.

The minimum value is equal to 0 and it is achieved at two points; two first coordinates of minimum points are X_{01}=(-0.9472, 0.9709) and X_{02}=(-1.0527, 1.0291) and others are equal to 1. The maximum value is larger than 0.459. B in (3) is re-scaled to a unit cube and two first coordinates the minimum points in new scales are (0.2368, 0.7572) and (0.2632, 0.7427) and others are equal to 0.75. The largest enclosure A_θ constitutes approximately 5% of B, while the estimate of the ratio is $vol(A)/vol(B) = 0.0005$; it is obtained from 10^6 random trials with uniform distribution over B. The sizes of enclosures for different θ are shown as dash-dot line (2) in Figure 5.

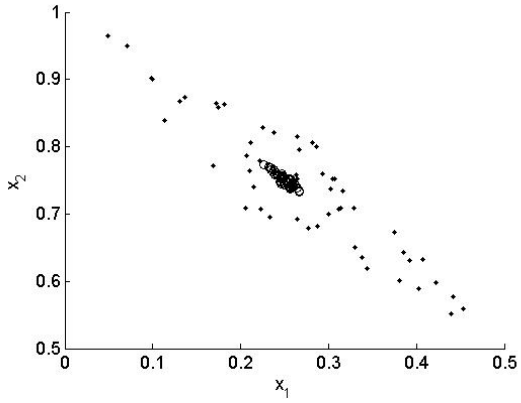

Figure 6. Rhomb wise feasible region of problem (3) is indicated by feasible points (o); points (.) belong to the enclosure with $\theta = 0$

3. Optimization algorithms

Two stage procedure is performed to solve optimization problems with small implicitly defined feasible regions: search for feasible region and optimization in the region approximated using found feasible points.

3.1. Generation of feasible points

Algorithm generating a set of points in feasible region A is a sequential procedure which starts with generating points in A_0 and finishes with $A = A_1$. The generation of points in A_θ is guided by information obtained from points in A_π, where $\pi < \theta$. An increase of the index implies a decrease in the size of the enclosure. The evolution of set of points is similar to the evolution of biological population driven by worsening environmental conditions. Developed three algorithms are based on this approach.

3.1.1. Algorithm I

The generation living in conditions defined by π produces descendants. The new generation is composed of the individuals of the old generation and their descendants who survive in the new conditions, defined by θ. The descendants are produced by multi-parent crossover aiming to ensure diversity in the new generation, where the diversity may be interpreted as the uniformity of the distribution of points in A_θ. The parents who would survive under the new conditions are chosen for crossover more frequently than those who would not; however those who would survive normally constitute only a small fraction of the old generation. For a set, J of j multi-parents, one survivable parent and $j-1$ non-survivable parents are chosen. The choice in corresponding subsets is random with uniform distribution. The crossover is

defined by formula

$$Y = \sum_{i \in J} \beta_i X_i ,\ \beta_i = \alpha_i + (1-\gamma)\left(\frac{1}{j} - \alpha_i\right),$$
$$\alpha_i \geq 0 ,\ \gamma \geq 1 ,\ \sum_{i \in J} \alpha_i = 1 ,$$

where α_i are weights; weight of the survivable parent is 0.5 and the others weights are generated randomly with uniform distribution. The coefficient $\gamma \geq 1$ defines central extension of the convex hull of points X_i with respect to the centre (average of the points). The crossover is a generalization of the convex (arithmetic) crossover for the multi-parent case [2]. The crossover of survivable and not survivable parents (points satisfying constrains and not satisfying constrains) is similar to the crossover used in GENOCOP [2].

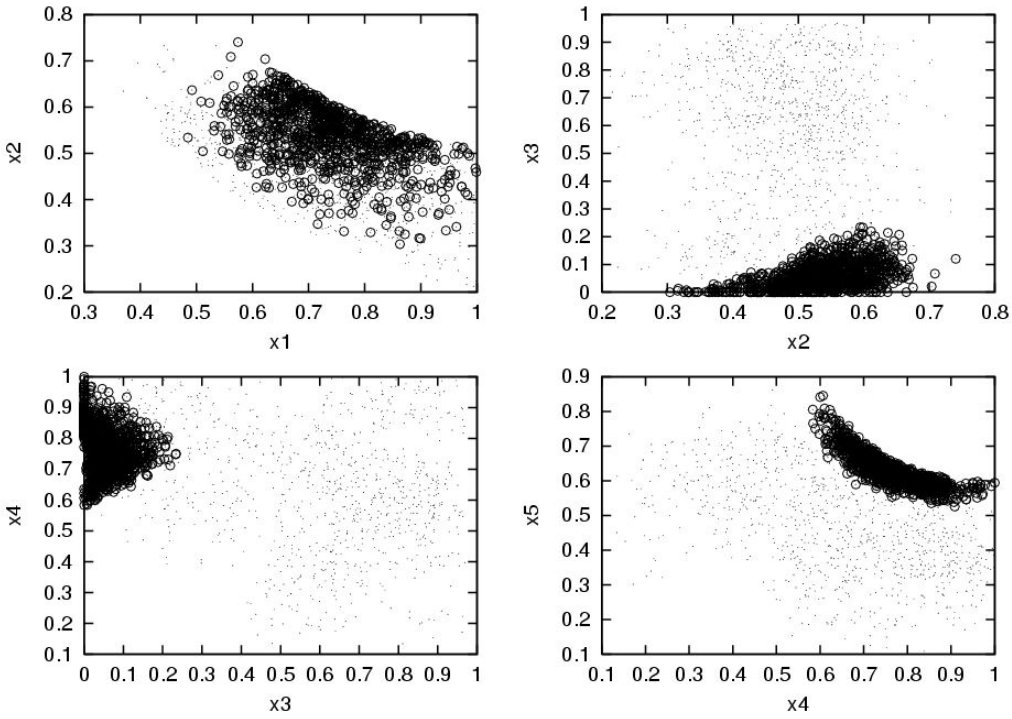

Figure 7. Projection of trial points to the plane of the variables pairs. Feasible points are indicated with large circles, and infeasible points with small dots

3.1.2. Algorithm II

Population living in conditions defined by π is used for finding new individuals living in conditions defined by θ. In order to collect a set of points in the enclosure θ, additional points are generated. New population consists of feasible points in the enclosure A_θ. New points are generated over the hypercube:

$$[x_i - \Delta; x_i + \Delta],\ i = 1, \ldots, n ,\ n = 5 ,\ \Delta = 0.05$$

around randomly chosen point from enclosure A_π. Δ was experimentally chosen and is equal to $(vol(A)/vol(B))*100$. m points are randomly generated in the defined hyper-rectangular. The size m was chosen experimentally and is equal $m = 2n$. New generated points are collected to new population if they are feasible in A_θ enclosure.

3.1.3. Algorithm III

Projection of trial points to the plane of the variables pairs in Figure 7 shows that some variables vary in large enough intervals and other variables vary in very small intervals of the hypercube. Therefore, to make new point by changing value of one variable in randomly chosen point may be purposeful.

Population living in conditions defined by π is used for finding new individuals living in conditions defined by θ. As in Algorithm II hypercube:

$$[x_i - \Delta; x_i + \Delta],\ i = 1, \ldots, n ,\ n = 5 ,\ \Delta = 0.05$$

around randomly chosen point from population living in conditions defined by π is defined. Randomly chosen variable's value is changed by a new value, which is randomly generated in the hypercube. Point, feasible in A_θ enclosure, is involved into new generation. Procedure is repeated m times, and this count, as in Algorithm II, is equal $m = 2n$. This cycle is repeated until N feasible points are collected.

3.1.4. Computational results

We aimed to compare the efficiency of the algorithms. One of the criteria chosen for comparison is the amount of points generated on the ground of

points in π enclosure for finding N points in θ enclosure. The efficiency of the algorithms could be compared using other criteria such as the average CPU Time of point's generation, the best of minimal values found, average of the minimal values found, standard deviation of the minimal values found and average number of function calls. Computations were made on a 700 MHz Pentium based computer with 256 MB of RAM.

The dependence of hypervolumes of enclosures on θ in Figure 5 demonstrates that hypervolumes decrease as the continuation parameter approximates to one. The values of the continuation parameter are chosen $\theta = 0.1k$, $k = 0,\ldots,10$.

Computations in each enclosure are terminated after the amount of points in θ enclosure comes to $N = 10^2$, when $\theta = 0$ and $N = 10^3$ when $\theta = 0.1,\ldots,1$. Ten independent runs are executed for each algorithm with different initial set of randomly generated points with uniform distribution over the hypercube. Table 1 contains the results of this experiment. The following notations are used: Alg. – the abbreviated algorithm's name; avgn – average number of points generated on the ground of the points in A_π for finding N points in A_θ, where $\pi = \theta - 0.1$; Total – sum of the average amount of points generated in sequential procedure, where $\theta = 0,0.1,0.2,\ldots,1$.

Table 2 lists the estimations of other criteria that are used to compare the algorithms efficiency.

According to computational results presented in Tables 1 and 2, the problem of finding feasible points is best solved by Algorithm II.

Table 1. The average amount of points in π enclosure used to generate N=1000 in each θ enclosure

Alg.	avgn $\theta=0.1$	avgn $\theta=0.2$	avgn $\theta=0.3$	avgn $\theta=0.4$	avgn $\theta=0.5$	avgn $\theta=0.6$	avgn $\theta=0.7$	avgn $\theta=0.8$	avgn $\theta=0.9$	avgn $\theta=1$	Total
Problem I											
Algorithm I	11392	1765	1760	2099	2153	2513	4101	4971	11054	1677887	1719695
Algorithm II	525	27	33	47	61	82	117	188	402	205997	207479
AlgorithmIII	304	304	307	306	313	329	335	353	432	246854	249837
Problem II											
Algorithm I	2306	1269	1312	1334	1470	1590	1725	1884	2437	1851	17178
Algorithm II	372	49	78	92	126	154	218	373	790	1632	3884
AlgorithmIII	405	381	388	409	445	510	654	814	1084	2875	7965
Industrial problem [3]											
Algorithm I	3817	1148	1227	1491	1565	1873	2514	3992	7784	10983	36394
Algorithm II	1354	1086	1077	1126	1189	1247	1603	2620	3621	6478	21401
AlgorithmIII	1371	1386	1270	1287	1324	1423	1782	2490	3755	7573	23661

Table 2. Criteria defining the algorithms efficiency

	Criteria	Algorithm I	Algorithm II	Algorithm III
Problem I (formula 2)	Average of the computational time (s)	869.5	74.34	102.82
	Best minimum	6.4419e-06	1.1283e-06	8.0762e-03
	Average of the minimal values found	0.02659	0.02584	0.03618
	Standard deviation of the minimal values found	0.025246	0.026647	0.018709
	Number of function calls	1719695	207479	249837
Problem II (formula 3)	Average of the computational time (s)	10.8	4.9	4.8
	Best minimum	0.034124	0.033858	0.034973
	Average of the minimal values found	0.246935	0.233409	0.235262
	Standard deviation of the minimal values found	0.099571	0.095470	0.096406
	Number of function calls	17178	3884	7965
Industrial problem [3]	Average of the computational time (s)	$3.04*10^3$	$2.54*10^3$	$2.73*10^3$
	Best minimum	-3.2184e+08	-3.2218e+08	-3.2104e+08
	Average of the minimal values found	-2.7453e+08	-2.7289e+08	-2.7069e+08
	Standard deviation of the minimal values found	1.1677e+07	1.3782e+07	1.1241e+07
	Number of function calls	36394	21401	23661

3.2 Optimization in feasible region

The proposed optimization algorithms are based on evolutionary technique. The trial points are generated by search algorithms modelling a population of individuals evolving under natural selection pressure hardening because of worsening environmental conditions. Selection is based on individual's fitness modelled by objective function value.

3.2.1. Adaptive search for optimum

The starting population consists of points in the feasible region A. The current population is described by the eigenvectors and eigenvalues of the covariance matrix of these points, and the average vector of these points. The trial points are modelled as realizations of a Gaussian random vector with average equal to $X_a - \varphi(X_{\min} - X_a)$, where X_a is the average of the current population, $X_{\min}$ is the best point found in the current population. A newly generated point is included in the new population if its function value is less than the average of the current population. The new population is formed of the best points of both the current population and newly generated points. The new population becomes current one, its eigenvalues and eigenvectors are calculated, and random generation of new points is repeated.

3.2.2. Evolutionary search for optimum

Descendants are produced according to the convex multiparent crossover

$$X_i = \alpha X_{\min} + \beta X_{r1} + \gamma X_{r2},\ i = 1...N, \qquad (4)$$

where $X_{\min}$ is the best point found, X_{r1} is a randomly selected point from $0.2N$ best points of the parent generation and X_{r2} is a randomly selected point from $0.8N$ worst points of the parent generation. The weights in (4) are chosen as follows: $\alpha = 0.7$, β, γ - are generated randomly with uniform distribution, $\alpha + \beta + \gamma = 1$. The number of descendants is equal to the population size N. The new generation consists of $0.2N$ best points of survived descendants and $0.8N$ best points of parent population.

3.2.3. GA - based search for optimum

A probabilistic selection is performed with the selection probability P_j based on individual's fitness value: $P_i = F_i \Big/ \sum_{j=1}^{N} F_j$, where F_i equals the fitness of individual i, N - population size. Individual i is selected if

$$C_{i-1} < U(0,1) \le C_i,$$

where $C_i = \sum_{j=1}^{i} P_j$ is the cumulated probability of the population. Each individual is coded into sequence of $bits*n$ binary digits, where n is variable number in individual and $bits = 16$. Two genetic operators were applied in this algorithm: multi-point crossover and mutation. Multi-point crossover takes two individuals and produces two new descendants. Two individuals are crossed from $((bits*i)-2)th$ and $(bits*i)th$ positions, where $i = 1,...,n$, with cross rate $p_c = 0.7$. Binary mutation flips each bit with probability $p_m = 0.001$ according to equation (5).

$$x_i' = \begin{cases} 1 - x_i, & if\ \ U(0,1) < p_m \\ x_i, & otherwise \end{cases}. \qquad (5)$$

Best descendants and best points from parent population constitute new population of N individuals.

3.2.4. Computational results

The goal of these experiments is to estimate the efficiency of the optimization algorithms. For comparison of the algorithms the following criteria can be used: best minimum found, the average of the found minimums, standard deviation of the found minimums, number of function calls used to obtain the best minimum value and computational time used to find best minimum value. The way of seeking minimum point is also an important criterion in algorithms comparison.

The population size is chosen 100, 300, 500 and generation's number is chosen 10, 20, 30, 40, and 50. Computations are terminated after the defined number of generations. 50 independent runs (each with different seed) are executed.

The results of the experiment are listed in Table 3. Best results for Problem I have presented an Evolutionary search algorithm. The best minimum, which is equal to 1.5001e-28, was obtained during the experiment with population size 500 and number of generation 30. Very close result to our known minimum point of this problem has carried an Adaptive search algorithm with best function value 3.4823e-011. For comparison of these two algorithms it is useful to mention that Adaptive search algorithm requires less function calls to find best point.

The best minimum for Problem II with rhomb wise feasible region was obtained during experiment with population size 500 and number of generations 40. Found minimum point is equal to known function minimizer and function value in this point is 0. Adaptive search algorithm has presented best results for Industrial problem [3]. The experiment with population size 100 and number of generations 40 produced the best result for the Industrial problem. The best minimum value found is -3.2542e+008 at the point (0.79774, 0.50082, 0.000084041, 0.84825, 0.54592). The average of found estimates of minimum is –3.2291e+008. Best points found by Evolutionary search and GA – based search algorithms also are acceptable, but averages of the found minimum

value's show that most of found best points are out of our target area.

The goal of another experimental investigation was to see how algorithms seek a minimum point. Experiment parameters and initial population were chosen the same, whereby best points of problems were found. In Figure 8 there are shown experimental results for three problems, which were solved with Adaptive search, Evolutionary search and GA based search algorithms. All algorithms start minimum search at the same initial population (the best value from initial population in Figure 8 is marked by circle). Starting at initial population in Industrial problem in experiment with population size 100 and number of generations 40, minimum is reached through 28 generations and in Problem I with rhomb wise feasible region, in experiment with population size 500 and number of generations 40, through 12 generations using Adaptive search algorithm. Evolutionary search algorithm in experiment with population size 500 and number of generations 30, finds best minimum through 30 generations in Problem II. Graph (3) in Figure 8 shows that Adaptive search finds a point close to the best minimum point

As seen in Figure 9 computational time depends on population size and objective function. In this experiment the fastest algorithm is Adaptive search. When population size grows up, the computational time of GA – based search algorithm grows up faster then other algorithms computational time.

Table 3. Criteria defining the algorithms efficiency

		Adaptive search	Evolution search	GA–based search
Problem I (formula 2)	Best minimum	3.4823e-011	1.5001e-028	4.5251e-007
	Average of the minimal values found	0.0047	3.5295e-006	6.2715e-005
	Standard deviation of the minimal values found	0.0102	1.7656e-005	5.7301e-005
	Number of function calls	11000	15000	12500
	Average of the computational time (s)	21	46.37	246
Problem II (formula 3)	Best minimum	0	7.4523e-005	0.0011
	Average of the minimal values found	0.0022	0.0323	0.0256
	Standard deviation of the minimal values found	0.0042	0.0166	0.0155
	Number of function calls	6000	20000	9500
	Average of the computational time (s)	13.2	73	233
Industrial problem [3]	Best minimum	-3.2542e+008	-3.2489e+008	-3.2479e+008
	Average of the minimal values found	-3.2291e+008	-3.1248e+008	-3.1938e+008
	Standard deviation of the minimal values found	2.1929e+006	4.2205e+007	4.4575e+006
	Number of function calls	2800	600	14700
	Average of the computational time (s)	44	19	260

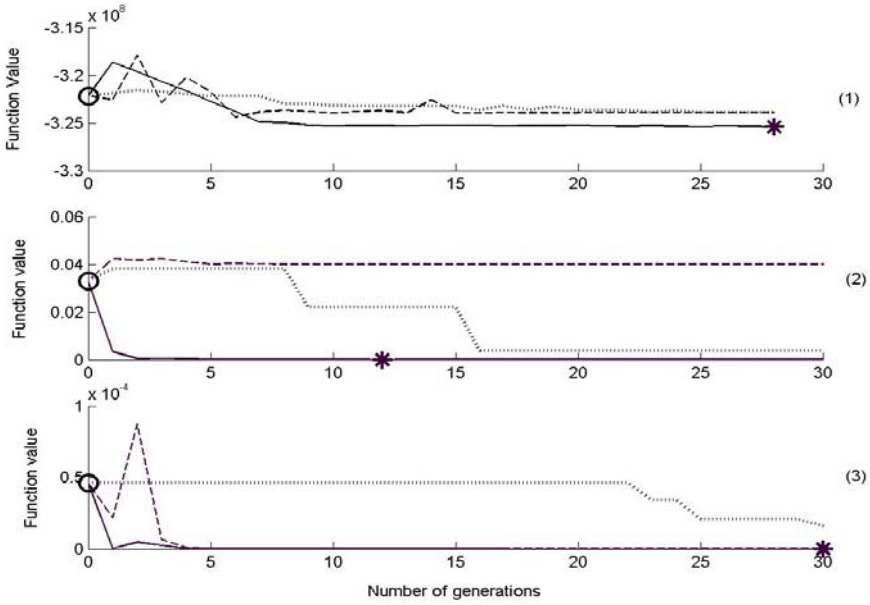

Figure 8. Algorithms comparison; (1) – industrial problem, (2) – Problem II, (3) – Problem I; * – best minimum value found; O – best value from initial population; Adaptive search – solid line; Evolutionary search – dashed line; GA – based search – dot line

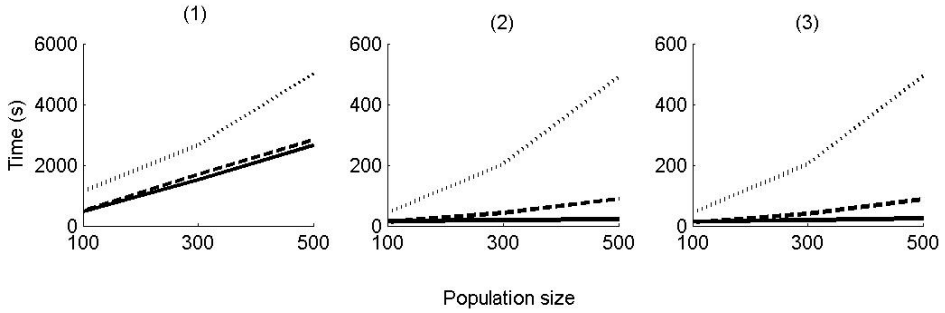

Figure 9. Computational time of 50 generations with population size (100, 300, 500) performed on minimization of three problems: (1) – industrial problem, (2) – Problem I, (3) – Problem II; Adaptive search time function– solid line; Evolutionary search time function – dashed line; GA – based search time function – dot line

4. Conclusions

Optimization problems with small implicitly defined feasible region are considered. Two stage procedure is used to solve such problems: search for feasible region and optimization in the region approximated using the found feasible points.

Three algorithms are proposed to find points in feasible region, and three algorithms are proposed to find minimum value and a minimizer in the approximated feasible region.

For the experimental investigation of algorithms efficiency a class of test problems is elaborated.

The results of experimental investigation show that Algorithm II is the best to locate feasible region. The best of three algorithms for optimization is the Adaptive search algorithm.

References

[1] **E.S. Fraga, M.A. Steffens, I.D.L. Boggle, A.K. Hind.** An object oriented framework for process synthesis and simulation. *In M.F. Malone, J.A. Trainham, and B. Carnahan, editors, Foundations of Computer-Aided Process Design, Vol.*95 *of AIChE Symposium Series*, 2000, 446-449.

[2] **Z. Michalevich.** Genetic Algorithms + Data Structure = Evolution Programs**.** *Springer, NY*, 1996.

[3] **A. Zilinskas, E.S. Fraga, A. Mackute, A. Varoneckas.** Adaptive search for optimum in a problem of oil stabilization process. *In I. Parmee, Adaptive Computing in Design and Manufacture VI, Computer-Aided Chemical Engineering, Springer, London*, 2004.

[4] **A. Zilinskas, A. Mackute,** On optimization over a small implicitly defined feasible region. *Proceedings of the* 15*th International Conference on Systems Science, Wroclaw*, 2004, 267-304.