Materiały VI Krajowej Konferencji Algorytmy Ewolucyjne i Optymalizacja Globalna 173-182
Łagów, 26-28 maja 2003

GOOL – GLOBAL OPTIMIZATION OBJECT-ORIENTED LIBRARY

Marek Publicewicz
Institute of Control and Computation Engineering, ul. Nowowiejska 15/19, 00-665 Warsaw
e-mail: m.publicewicz@elka.pw.edu.pl

Ewa Niewiadomska-Szynkiewicz
Institute of Control and Computation Engineering, ul. Nowowiejska 15/19, 00-665 Warsaw
Research Academic Computer Network (NASK), ul. Wąwozowa 18, 02-796 Warsaw
e-mail: e-n-s@ia.pw.edu.pl

Abstract

The paper presents an integrated environment, called GOOL (*Global Optimization Object-oriented Library*), which can be used to solve complex problems of searching for the global minimum (maximum) of performance function. It provides a framework that allows to perform numerical experiments. GOOL offers the graphical environment for supporting the considered problem definition and calculation results presentation. The main component of the GOOL system is the library of random search generators and optimization algorithms for convex and nonconvex, unconstrained and constrained problems. The numerical results for the test problem are presented in the final part of the article.

Key words

global unconstrained optimization, global constrained optimization, random generators, software systems

I. INTRODUCTION

Most of the existing libraries of optimization techniques focus on the problem of computing locally optimal solutions [2]. Global optimization is concerned with the computation and characterization of global optima of nonconvex function. Many of the real-world optimization problems require use of heuristic methods which do not guarantee the optimum solution, but rather provide with the reasonable solution in a reasonable time. During last decades many theoretical, algorithmic and computational contributions helped to solve multiextreme problems arising from practical applications.

The paper describes the organization, ability and usage of an integrated graphical software environment, called GOOL (*Global Optimization Object-oriented Library*), which provides a library of methods for solving the following very general problem:

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

$$\text{subject to:} \quad g_i(x) \le 0 \qquad i = 1,...,m$$

where f and g_i are real-valued functions defined on the domain $X \subseteq \Re^n$.

The paper is organized as follows. First, the short description of the system is presented. Next, the principle features of the deterministic and nondeterministic global optimization methods provided in GOOL are briefly discussed. Finally, the comparative study of the implemented optimization techniques is presented.

II. DESCRIPTION OF THE GOOL SYSTEM

A. System structure

The first version of the system, called VSO (*Visual System for Optimization*) was described in [6]. GOOL is much more advanced and has wider range of applications. Two versions of the system are offered:

- GOOL/COM –software for large scale practical optimization problems,
- GOOL/VIS – software for educational purposes and research.

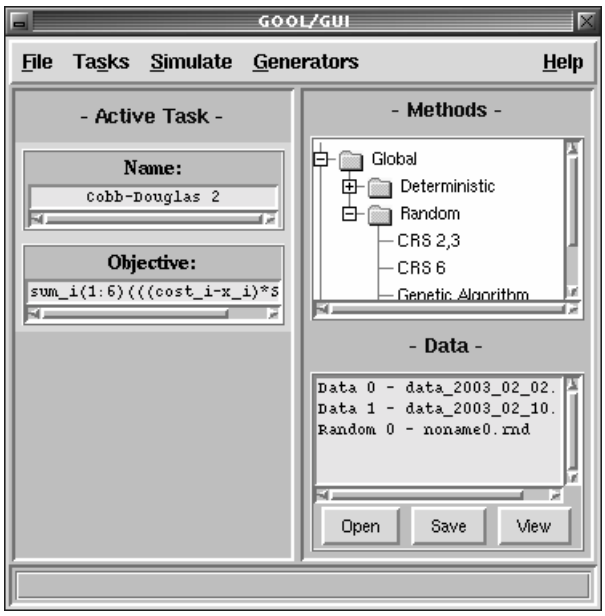

Fig.1 GOOL – the main window

Two main components of GOOL may be distinguish: library of numerical methods (random generators and optimization algorithms) and user interface. The system facilities are provided in the form of four groups of services. These are:

- *user interface services*, which provide a consistent user interface. The most important tasks of the user interface are as follows: supporting the process of defining a considered optimization problem, presenting of the calculation results, providing communication with the user (during problem definition and calculation process). The interface is graphical;
- *calculation management services*, which manage execution of chosen optimization methods;
- *communication management services*, which manage communications between calculation process and user interface;
- *data repository services*, which provide a store for all data objects: all defined options, parameters and calculation results.

The system may operate in two modes:

- *research mode* – only the results of all iterations are displayed,
- *educational mode* – the detailed report of each iteration is presented (new point, new search direction, etc.), see Fig. 2.

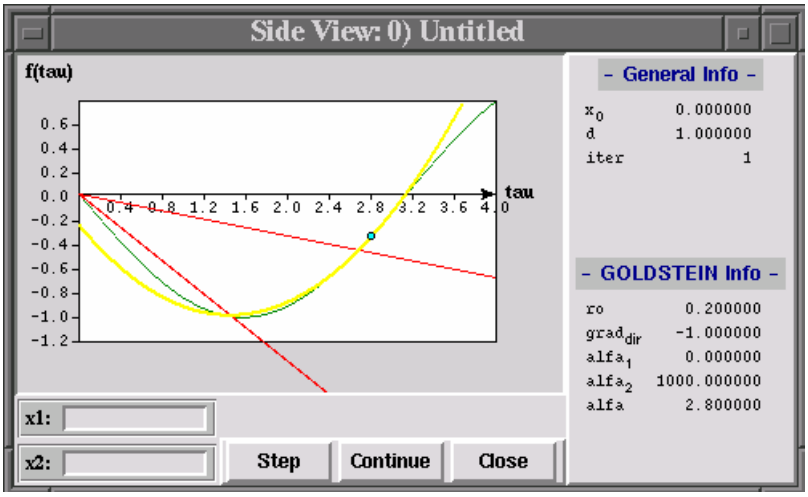

Fig.2 Educational mode

B. Optimization problem definition

There are two possibilities of the considered task definition. The first is to enter the objective function together with all constraints, using the GOOL editor (see Fig.4). The symbolic expressions analyzer allows to enter quite complicated functions concerning such expressions like: *pow*, *sin*, *sum*, etc., and *iterative expressions*. The gradient is calculated if necessary. The second possibility is dedicated to the complex optimization problems, where values of the objective function are calculated based on simulation (*simulation-optimization* scheme). In this approach it is assumed that the methods from the library provide decision variables and receive the values of the objective function from the user application.

C. Results visualization

The system provides tools for on-line monitoring of the computed results. The following graphical presentation techniques are available: 2D, 3D graphs (see Fig. 6, 3), leaves (see Fig. 7) of the function and a table of numbers. The visualization of a multidimensional problem is achieved by displaying in the separate windows the leaves for each pair of variables, under the assumption that all other variables are fixed. The results presentation is organized in different ways, and is fitted to the optimization method (points, lines, grids). Multiple windows with the results for different range of data can be active during one run of the program. The user can chose options of presentation (zoom, colors, results of many optimizations in one window, etc.).

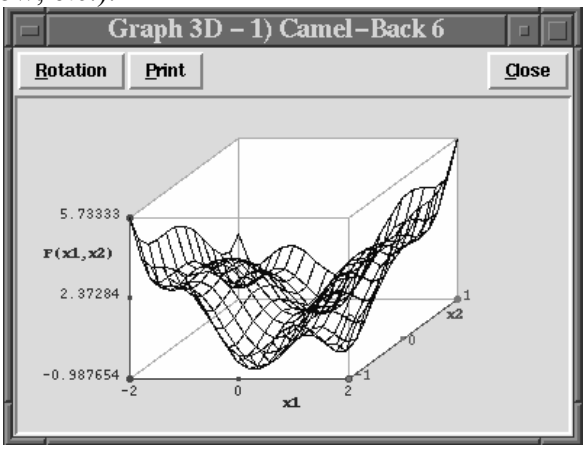

Fig.3 3D presentation

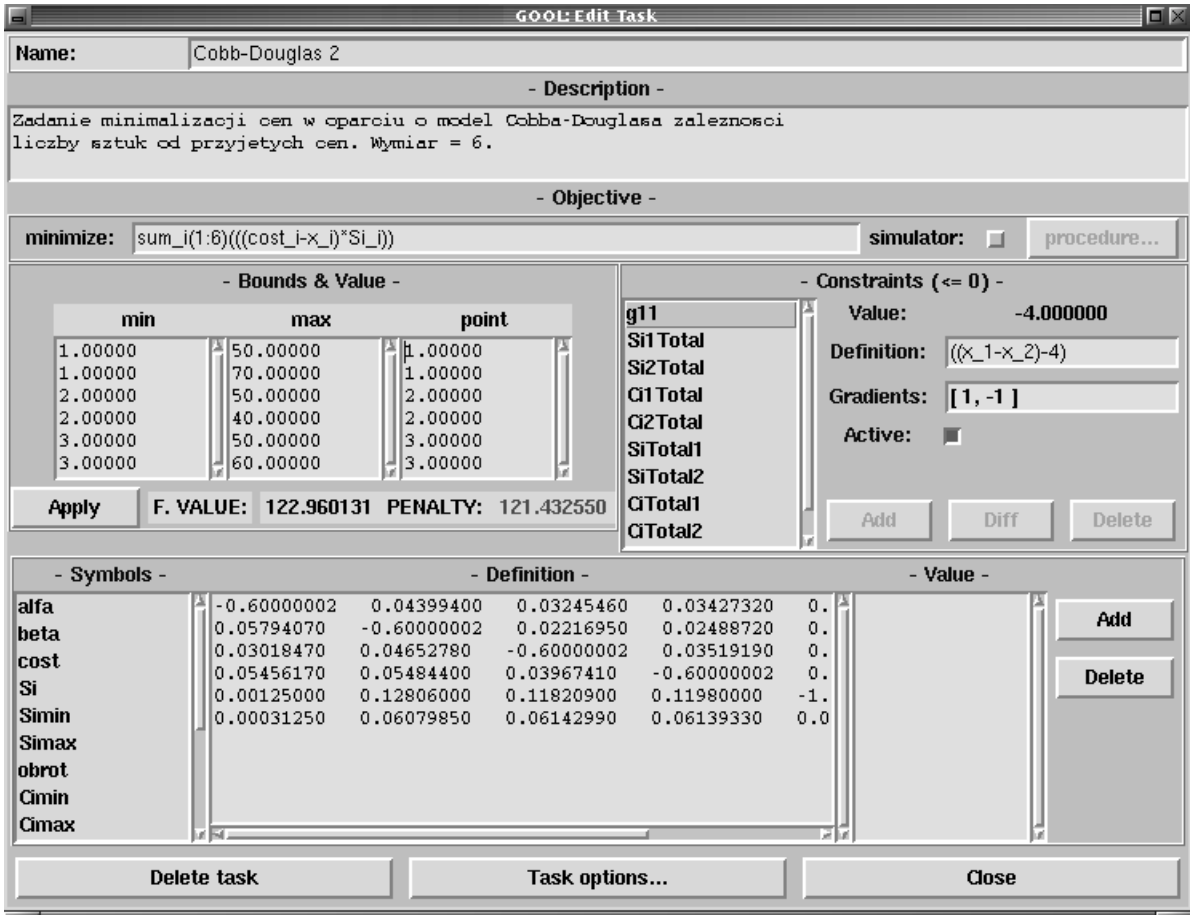

Fig.4 The optimization problem formulating

D. The program operation

The interaction with GOOL is organized as follows. At the beginning the user is asked to define the problem to be solved. The library of several well known global optimization problems is provided in GOOL. The user can chose one of them or enter a new one. Next, the optimization method has to be selected. Within the next step the user is asked to provide some information related to the considered method and calculation process. This information includes: parameters typical for the considered algorithm, type of the stop criterion, maximal number of iterations, type of results visualization, etc.

After assigning all information the calculations are performed. The user employs monitoring and analysis of the current situation. It helps him to assess the effectiveness of the chosen optimization algorithm. The calculations may be interrupted.

E. Implementation details

GOOL may operate under MS-Windows, Windows-NT and Linux operating systems. Two different languages were applied to its implementation: library of numerical methods is written in C++ and user interface in Tcl/Tk. Two basic classes were define: *generator* for random numbers generators implementation and *algorithm* for optimization methods. The library may be easily extended by new methods developed by the user.

III. RANDOM NUMBER GENERATORS

Many heuristic algorithms provided in GOOL use random number generators for new point calculation. The large number of random generators have been developed over the last decades. Several procedures representing different types of generators are available in the library:

- uniform - two variants,
- normal (Gaussian) – three variants,
- Beta distribution,
- Cauchy distribution.

Sequences of *n*-tuples that fill *n*-space more uniformly than uncorrelated random points are called quasi-random sequences. That term is somewhat of a misnomer, since there is nothing "random" about quasi-random sequences – they are cleverly crafted to be, in fact sub-random. Three such sequences are available in GOOL:

- Halton,
- Sobol,
- Faure.

The results obtained for Faure sequence is presented in Fig. 5.

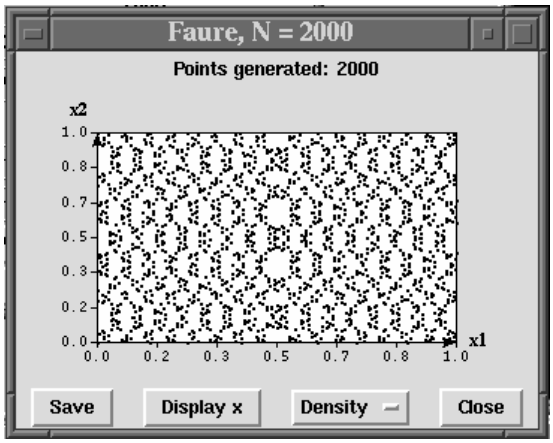

Fig.5 Faure sequence, N=2000

IV. OPTIMIZATION ALGORITHMS

The GOOL library provides different optimization techniques for calculating local and global minimum. The focus is on the algorithms designed for nonconvex optimization.

A. Local optimization

Several techniques for calculating local minimum of the performance index were implemented in GOOL. The following methods for one-dimensional search are available [2, 7]:

- Golden Section Search,
- Parabolic Interpolation,
- One-dimensional Search with First Derivatives (Goldstein test)

Two well known nongradient optimization methods in multidimensions [2, 7]:

- Downhill Simplex Algorithm (Nelder-Mead),

- Direction Set (Powell's) Method.

B. Global optimization

Few methods of global optimization were implemented in the GOOL library. We can distinguish two groups: deterministic and nondeterministc. In the case of deterministic approach five algorithms were considered: four grid algorithms and one trajectory method. In the case of nondeterministic, the controlled random search, population and genetic algorithms were taken into account.

Grid methods

Practical objective functions normally have bounded rate of change. Therefore the search on a sufficiently dense grid guarantees the detection of the global minimum with the prescribed accuracy. In the case of grid methods (covering methods) the assumption which is made is that the slopes of f and g_i in (1) are bounded. In such case, these functions are said to be *Lipschitz*, and satisfy the condition:

$$\exists_{L>0} \quad \forall_{x,y\in X} \quad |f(x)-f(y)| \leq L \|x-y\| \tag{2}$$

where L is the *Lipschitz* constant.

The uniform covering method is implemented in GOOL. Since the rate of change of the objective function is supposed to be bounded, the grid does not have to be uniform. It seems more rational to have density of nodes in subregions with small function values and vice versa. There are some approaches to construct such non-uniform grids, some of them are offered in GOOL: Pijavskij's algorithm for one-dimensional case [5, 10], (see Fig. 6) and three algorithms for multidimensional case [5]: Galperin, Gourdin-Hansen-Jeaumard and Meewella-Mayne. The most advanced is Meewella-Mayne method of linear sub-approximations of the considered performance function f.

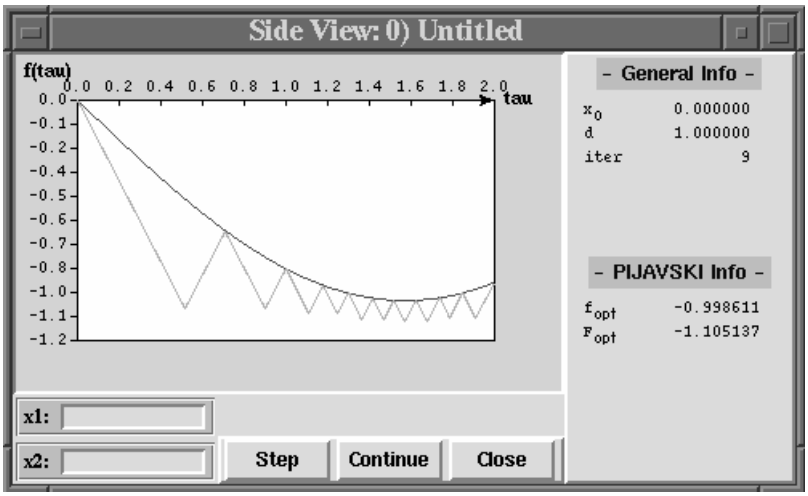

Fig.6. Results for Pijavskij's algorithm

Trajectory method

Several algorithms generalize or modify the local minimization methods for the case of the global minimization. It can be done by modifying the differential equations describing the local descent trajectory. The most important property of these equations is that the trajectory passes through the neighbourhood of a majority of stationary points of the objective function. The Griewank's algorithm [4, 10] of chaotic movement implemented in GOOL belongs to this group.

Random search

Three controlled random search algorithms are available: CRS2, CRS3 [8] and CRS6. In principle, they were designed as a combination of local optimization algorithm with the global search procedure. The main idea of CRS2 and CRS3 is to transform n+1 dimension randomly generated simplex. In the case of CRS6 β-random generator is applied to generate a new point.

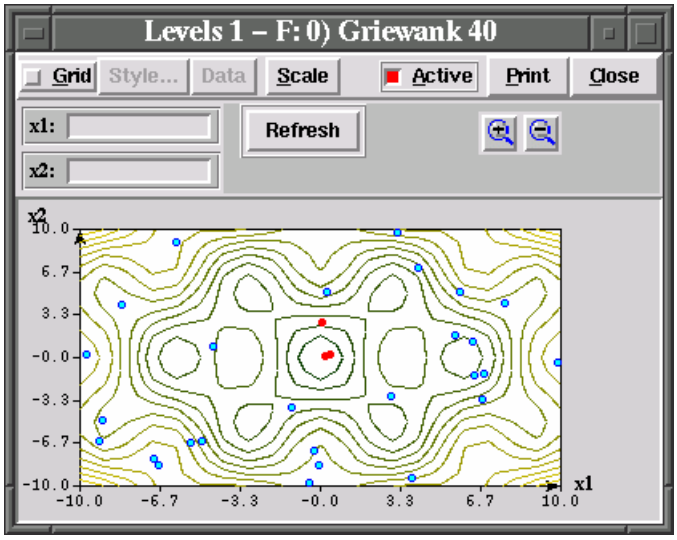

Fig.7. Results for CRS methods

Simulated annealing (SA)

Simulated annealing is a stochastic method applying a probabilistic mechanism that enables search procedures to escape from local minima. This is done by accepting, in addition to transitions corresponding to a decrease in function value, transitions corresponding to an increase in function value. The latter is done in a limited way by means of a stochastic acceptance criterion. The probability of accepting deteriorations descends slowly towards zero. This procedure will lead to a global minimum. At the hart of this method is a analogy with thermodynamics, specifically with the way that liquids freeze and crystallize, or metals cool and anneal. The amazing fact is that, for slowly cooled systems, nature is able to find the minimum energy state. The GOOL library provides the implementation of SA as described in [1].

Clustering methods

Clustering methods is a class of global optimization methods, which as an important part include a claster analysis technique. For grouping points around minima different strategies have been used. One of them is based on the idea that retaining only points with relatively low function values these points would form groups around some of the local minima. Another strategy is to push each point towards a local minimum by performing a few steps of a local method. The algorithm developed by Törn [10], (with different grouping techniques) is provided in our library.

Evolution-based methods

The evolution techniques [3] are probabilistic algorithms which maintain a population of individuals (points) for each iteration. Each individual represents a potential solution to the problem. Each solution is evaluated to give some measure of its "fitness". Then, a new population is formed by selecting the more fit individuals. Some members of the new population is transformed by "genetic" operators to form new solutions. Two versions of

algorithms from this group are offered in GOOL: classical genetic algorithm using fixed-length binary strings for its individuals and real-valued individuals.

V. NUMERICAL EXPERIMENTS

Numerical experiments were performed for the following optimization problem:

$$\min_x \left(f(x) = 0.5 - \sqrt{\frac{x_1^2 + x_2^2}{x_1^2 + (1 - x_2)^2}} \right)$$

under the constraints:

$$g_1(x) = \left(x_1 - \frac{2\sqrt{2}}{3} \right)^2 + \left(x_2 - \frac{\sqrt{2} - 1}{3} \right)^2 - 0.22 \geq 0$$

$$g_2(x) = x_1^2 + \left(x_2 - \frac{\sqrt{2} - 1}{3} \right)^2 - 0.22 \geq 0$$

$$0 \leq x_1 \leq 0.6, \quad 0 \leq x_2 \leq \frac{\sqrt{2} - 1}{3}$$

The results are given in the Table 1. The values collected in the adequate columns represents: (x_1, x_2) - problem solution, $f(x)$ – value of the objective function in the problem solution, *time* – time of calculations, *f_call* – number of the objective function evaluations, *hits* - number of calculated infeasible points (during the algorithm run).

method	x_1	x_2	$f(x)$	*time[s]*	*f_call*	*hits*
GA	0.473676	0.13807	-0.001661	7	50051	47362
SA	0.473353	0.13807	-0.001425	4	10384	8907
CRS2	0.473768	0.13806	-0.001727	3	632	389
CRS3	0.473767	0.13807	-0.001728	3	728	385
CRS6	0.473724	0.13804	-0.001671	1	628	185
Törn	0.473767	0.13807	-0.001728	3	3410	1065

Table 1. Test results for the testing problem.

In the tests, CRS2, CRS3 and Törn methods were able to find the better solution than the other methods. The times of calculations were the shortest too. The worst result was obtained for SA method. In the case of GA algorithm the convergence to the global minimum was far slower, which was caused by many infeasible individuals computed during the optimization process.

However, it should be pointed that the considered testing problem has a very specific structure – the solution is in the irregular point, just on the constraints. The tests performed for local optimization methods (SQP and cutting plane algorithms) failed.

VI. CONCLUSIONS

As a final conclusion we can say that the presented software system GOOL is suitable to solve different optimization problems and can be successfully used for global minimum calculating. The user-friendly interface allows to perform the numerical experiments in the effective manner both for research and education. It should be pointed that the package may be easily extended by new algorithms, which are specific to a chosen case study.

Acknowledgments. This work was supported by Research and Academic Computer Network (NASK) and the Polish Committee of Scientific Research under grant 7T11A 022 20.

The GOOL library is available at the web page **http://www.ia.pw.edu.pl/~ens/gool**

VI. BIBLIOGRAPHY

[1] Dekkers, A.,. Aarts, E., *Global optimization and simulated annealing*. Mathematical Programming, vol. 50, 1991.

[2] Findeisen, W., J. Szymanowski, A. Wierzbicki, *Teoria i metody obliczeniowe optymalizacji*. PWN, Warszawa. 1980.

[3] Goldberg, D., E., *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison–Wesley Pub. Co., 1989.

[4] Griewank, A., O., *Generalized Descent for Global Optimization*. Journal of Optimization Theory and Applications, Vol. 34, No.,2, str. 11-39, 1981.

[5] Horst, R., Pardalos, P.M., *Handbook of Global Optimization,* Kluwer, 1995.

[6] Ewa Niewiadomska-Szynkiewicz, Piotr Bolek, Robert Śliwiński, *Środowisko graficzne do badania metod optymalizacji globalnej*, Proc. of Evolutionary Algorithms and Global Optimization, 1999.

[7] Press, W.H., S. A. Tukolsky, W. T. Vetterling, B. P. Flannery, *Numerical Recipes in C. The Art of Scientific Computing*. Cambridge University Press. 1992.

[8] Price, W.L., *Global Optimization by Controlled Random Search*. JOTA, vol. 40, No. 3, July 1983.

[9] Rogers, J.W., Jr. R. A. Donnelly, *A Search Technique for Global Optimization in Chaotic Environment*. JOTA, vol. 61, No. 1, April 1989.

[10] Törn, A., Žilinskas, A., *Global Optimization*, Lecture Notes in Computer Science, Springer-Verlag, 1989.