SIMULATION ANALYSIS OF GLOBAL OPTIMIZATION ALGORITHMS AS TOOLS FOR SOLVING CHANCE CONSTRAINED PROGRAMMING PROBLEMS

Andrzej Z. GRZYBOWSKI
Institute of Mathematics, Faculty of Mechanical Engineering and Computer Science,
Częstochowa University of Technology, Dabrowskiego 69, 42-201 Czestochowa, Poland, e-mail: andrzej.grzybowski@im.pcz.pl

ABSTRACT

In the paper a chance constrained linear programming problem is considered in the case of join chance constraints with random both left and right hand sides. It is assumed that due to its complex stochastic nature the problem cannot be reduced to any equivalent deterministic problem. In such a case a Monte Carlo method combined with Global Optimization (GO) algorithms are proposed to solve the problem. A performance of various types of GO algorithms as tools for solving such problems are compared via computer simulations. The simulation results are presented and discussed in the paper.

Keywords: *chance constrained programming, Monte Carlo simulations, stochastic search, evolutionary algorithms, annealing type algorithms.*

1. INTRODUCTION

Chance Constrained Programming (CCP) or, more generally, stochastic programming deals with a class of optimization models and algorithms in which some of the data may be subject to significant uncertainty. Such models are appropriate when data cannot be measured/stated without error or when some parameters of the decision process are random in nature. The concept of CCP was introduced in the classical work of Charnes and Cooper [1]. Now CCP belongs to the major approaches for dealing with random parameters in optimization problems. Typical areas of application are engineering design applications [12], finance (e.g.[13]), budgeting ([2]) or portfolio analysis [5]. In models built for such real-world problems uncertainties like price of a final product, product demand, transportation costs, demographic conditions, currency exchange rates, rates of return etc. enter the inequalities describing the natural constraints that should be satisfied for proper working of a system under consideration.

Stochastic optimization problems belong to the most difficult problems of mathematical programming. It is because most of the existing computational methods are applicable only to convex problems. There are, however, many important applied optimization problems which are, at the same time, stochastic and non-convex. Many of them are also multi-extremal. Discussion of various computational aspect of CCP problems can be found in papers [5,8,10,11] or textbooks [4,7].

This paper is devoted to the linear programming problems which contain random parameters. It is assumed that due to their complex stochastic nature the problems cannot be reduced to any equivalent deterministic problems. To find the "best solution" it is proposed to make use of global optimization algorithms. However, it results from the statement of the problem, that the criterion function cannot be expressed by any closed-form mathematical expression and the value of the criterion functions can only be computed for each specific vector of decision variables. As a consequence we have to confine ourselves to gradient-free optimization methods. In presented studies GO methods based on the idea of the stochastic search are analyzed as tools for solving CCP problems. Such methods require only few assumptions about the underlying objective functions - they have so-called "black box" character, see [11,12,14]. In the approach presented in this paper Monte Carlo simulations are used for evaluation of the criterion function values, and thus they are the primary source of input information during the search process. It is very important feature of the examined problems because algorithms that aim at finding a global optimum usually have to evaluate the function many times. Consequently, the efficiency of GO algorithms in the considered case should be primarily measured by the number of function evaluations that has the main impact on the calculation time that it requires per iteration.

In this paper we compare the performance of the most popular stochastic global optimization methods: genetic algorithms, evolutionary search with soft selection and simulated annealing. The performance of the methods as a tools for solving the stochastic linear programming tasks is studied via computer simulations under two different utility criteria.

2. CHANCE CONSTRAINED LINEAR PROGRAMMING PROBLEM

Let us consider a classical (deterministic) linear programming problem:

$$\text{maximize} \quad f(x_1, \ldots, x_n) = c_1x_1+c_2x_2+ \ldots+c_nx_n$$

subject to the constrains (s.t.):

$$a_{i1}x_1+a_{i2}x_2+ \ldots+a_{in}x_n \le b_i \qquad i=1,\ldots,m$$
$$x_1\ge 0, \ldots, x_n\ge 0$$

where f is the objective function, $\mathbf{x}=[x_1, x_2, \ldots,x_n]^T$ is the decision variable vector, $\mathbf{A}=[a_{ij}]_{mxn}$ is the matrix of coefficients of the system of linear inequalities, a coefficient vector $\mathbf{b}=[b_1, b_2, \ldots,b_m]^T$ will be addressed as a right hand side of the constraints system, $\mathbf{c}=[c_1, c_2, \ldots,c_n]^T$ is a vector of the objective function coefficients.

As we have mention before, in many applications the elements of the tuple (**A,b,c**) cannot be considered as known constants. All or part of them are uncertain. Thus it is impossible to know which solution will appear to be feasible. In such cases, one would rather insist on decisions guaranteeing feasibility 'as much as possible'. It is justified by the fact that in such cases constraint violation can almost never be avoided because of unexpected random events. On the other hand, it makes sense to call decisions feasible (in a statistical meaning) whenever they are feasible with high probability, i.e., only a low percentage of realizations of the random parameters leads to constraint violation under this given decision. Such statistical concept of feasibility leads to CCP formulation of the problem, where the deterministic *constraints* are replaced with a *probabilistic* or *chance ones* in the following way:

$$\begin{aligned} &\text{maximize} \quad Ef(x_1, \ldots, x_n) = \mathrm{E}(c_1)x_1+\mathrm{E}(c_2)x_2+ \ldots+\mathrm{E}(c_n)x_n \\ &\text{s.t.} \\ &\qquad \Pr(a_{i1}x_1+a_{i2}x_2+ \ldots+a_{in}x_n \le b_i,\ i=1,\ldots,m) \ge q \\ &\qquad x_1 \ge 0, \ldots, x_n \ge 0 \end{aligned}$$

where $q \in [0,1]$ is a prescribed probability level.

The value of probability level is chosen by the decision maker in order to model the safety requirements. Sometimes, the probability level is strictly fixed from the very beginning (e.g., q=0.95,0.99 etc.). In other situations, the decision maker may only have a vague idea of a properly chosen level. It is obvious that higher values of q lead to fewer feasible decisions **x**, and hence to smaller optimal values of expected gain. In some simple cases (especially in case of individual chance constraints) the problem can be replaced with its deterministic equivalent, see e.g. [4, 7].

The main challenge in designing algorithms for general stochastic programming problems arises from the need to calculate conditional expectation and/or probability associated with multi-dimensional random variables. This make the CCP problems most difficult problems of mathematical programming. The computational challenges and methods in the field of optimization under uncertainty are addressed e.g. in [4],[7] and [10]. In this paper a situation is considered where, due to assumed complex stochastic nature of the problem, no deterministic equivalent is available. In order to find satisfactory stochastically feasible solution we propose two criteria, one leading to maximization of the probability of feasibility, and a second based on expected utility of a given solution. Then we compare various global optimization algorithms as tools for solving such problems.

3. PROBLEM DESCRIPTION

In the presented studies linear programming problems are examined in the case where all parameters determining the optimization task, i.e. the matrix **A** and vectors **b**, **c**, are random. It is assumed that the following expectations exist: E(**A**)=$\mathbf{\Lambda}$, E(**b**)=$\boldsymbol{\beta}$, E(**c**)=$\boldsymbol{\chi}$. In the sequel such a problem will be denoted CCLP(**A,b,c**). The performance of solutions found by a given GO method for the CCLP(**A,b,c**) problem is compared with the performance of the optimal solution found for the deterministic linear programming problem given by the parameters $\mathbf{\Lambda}$, $\boldsymbol{\beta}$, $\boldsymbol{\chi}$ - in the sequel the latter problem will be denoted as DLP($\mathbf{\Lambda}$, $\boldsymbol{\beta}$, $\boldsymbol{\chi}$).

The decision-maker dealing with the CCLP($\mathbf{\Lambda}$, $\boldsymbol{\beta}$, $\boldsymbol{\chi}$) problem should maximize both the probability q that a given systems of constraints will be satisfied and the expected value of the objective function. However, as it already emphasized, the two goals often appear to be contradictory (at least to some extent). Thus in our studies we use two simulation based indices of performance of a given solution. The first one is simply the estimated *probability of feasibility* P_f(**x**) i.e. the probability that the system of constraints will be satisfied when one use a given solution **x**. This index is given by the formula:

$$P_f(\mathbf{x}) = \frac{N_s}{N_{MC}} \tag{1}$$

where N_{MC} is a number of i.i.d. Monte Carlo realizations of CCLP(**A,b,c**) generated during the simulation process, N_s is the number of successful realizations (i.e. the realizations for which the system of constraints was satisfied).

In the simulations a single random realization of CCLP(**A,b,c**) is the realization of a random tuple (**A,b,c**) satisfying the condition E(**A**)=$\mathbf{\Lambda}$, E(**b**)=$\boldsymbol{\beta}$, E(**c**)=$\boldsymbol{\chi}$. It is additionally assumed that each element the random matrix **A** and vectors **b, c** has normal probability distribution with standard deviations equal to 10% of its absolute values.

The index P_f(**.**) should be used in all these cases where the negative consequences of the constraint violations are much more serious than the profits resulting from the increment of the criterion function.

The second criterion considered in this study takes into account both, the estimated probability P_f(**x**) as well as the expected value of the criterion function in the case when all constraints are satisfied. It is given by the following formula:

$$SIP(\mathbf{x}) = P_f(\mathbf{x}) \cdot \frac{1}{N_s} \sum_{i \in S} f_i(\mathbf{x}) \tag{2}$$

where S is the set of successful realizations indices, f_i(**x**) is the value of the objective function f obtained in the i-th successful realization of the problem, $i \in S$.

Without loss of generality, in the simulation study we confine ourselves to problems with positive objective functions.

4. GRADIENT-FREE GLOBAL OPTIMIZATION ALGORITHMS

"Evolutionary methods" (EM) is a general term that is used to refer to population-based metaheuristic optimization algorithms that use biology-inspired mechanisms like mutation, crossover, natural selection in order to refine a set of solution candidates iteratively. EM are perhaps the most popular search methods used for the global optimization tasks. All EM algorithms are

computer-based approximate representations of natural evolution. In these types of algorithms the population (of solutions) is altered over a sequence of generations according to stochastic analogues of the processes of evolution. In literature the EM algorithms are divided into two main groups: *genetic* algorithms and *evolutionary programming* methods.

Genetic algorithms (GAs) are a subclass of evolutionary algorithms where the elements of the search space are binary strings, (sometimes called genotypes, chromosomes). The genotypes are used in the reproduction operations whereas the values of the objective functions (so-called *fitness*) are computed on basis of the phenotypes in the problem space which are obtained via the genotype-phenotype mapping. The algorithm implemented in our simulations can be described as follows, see e.g. [12]:

Step 0 (Initialization) Set the initial population of K vectors $\boldsymbol{x}_i \in R^n$, $i=1,2,\ldots,K$ (phenotypes) and evaluate the fitness function for each of the vectors.
Step 1 (Parent selection) Select with replacement K parents from the full population. The parents are selected according to their fitness, with those phenotypes having a higher fitness value being selected more often.
Step 2 (Crossover) For each pair of parents identified in Step 1, perform crossover on the parents chromosomes at a randomly (uniformly) chosen splice point.
Step 3(Replacement and mutation) Replace the K chromosomes of current population of parents with the chromosomes of current population of offspring from Step 2. Then mutate the individual bits with uniform probability
Step 4 (Fitness and end test) Compute the fitness values for the population of N phenotypes corresponding to new chromosomes. Terminate the algorithm if the stopping criterion is met; else return to Step 1.
Step 5. Return the so far best generation and the fitness of its elements

While the GAs have traditionally relied on bit coding, the evolutionary programming methods (EP) have operated directly with the floating-point representations. In difference to other types of evolutionary algorithms, in evolutionary programming, a solution candidate is thought of as a phenotype (species) itself (using the EM terminology). Thus, selection and mutation are the only operators used in EP and recombination (crossover) is usually not applied., see [14]. There are various implementations of the idea in literature. In the simulations the performance so-called algorithm of the evolutionary search with soft selection (ES-SS) was examined. The algorithm implemented in our simulations is as follows, see e.g [3], [6]:

Step 0. Set the initial parent population of K vectors $\boldsymbol{x}_i \in R^n$, $i=1,2,\ldots,K$.
Step 1. Assign to each vector $\boldsymbol{x}_i$ $,i=1,\ldots,K,$ its *fitness* i.e. the value of the criterion $F(\boldsymbol{x}_i)$.
Step 2. Select parent $\boldsymbol{v}$ by *soft selection* i.e. with probability proportional to the its *fitness*.
Step 3.Create a descendant $\boldsymbol{w}$ from the chosen parent $\boldsymbol{x}$ by its random mutation: $\boldsymbol{w}=\boldsymbol{v}+\boldsymbol{Z}$, where $\boldsymbol{Z}$ is a random n-dimensional vector with coefficients having expected value equal to zero and given standard deviation σ_Z.
Step 4. Repeat steps 2 and 3 for K times to create a new K-element generation of n-dimensional vectors (so-called descendants)
Step 5. Replace the parent population with the descendant population
Step 6. Repeat the second to sixth steps until the stopping criterion is met
Step 7. Return the best element found and its fitness.

In our study a parent was selected with probability proportional to its relative fitness RF given by the formula:

$$RF(\mathbf{x}) = 99\frac{F(\mathbf{x}) - F_{\min}}{F_{\max} - F_{\min}} + 1$$

The initial population in both above presented algorithms was generated as a population of mutations of the optimal solution of the related DLP(Λ, β, χ) problem.

The third method adopted in our studies is the Simulated Annealing Algorithm (SA). It is perhaps historically first global optimization method based on stochastic search idea. It was developed by Kirkpatrick in the early 1980s although the main idea was introduced by Metropolis in 1953, [12]. In difference to the previously described EM, this algorithm does not use any biology-inspired mechanisms. The underlying idea is adapted from metallurgy and material science. Annealing is a heat treatment of material with the goal of altering its properties such as hardness. Metal crystals have small defects, dislocations of ions which weaken the overall structure. By heating the metal, the energy of the ions and, thus, their diffusion rate is increased. Then, the dislocations can be destroyed and the structure of the crystal is reformed as the material cools down and approaches its equilibrium state. When annealing metal, the initial temperature must not be too low and the cooling must be done sufficiently slowly so as to avoid the system getting stuck in a meta-stable, non-crystalline, state representing a local minimum of energy.

For the global optimization purpose the idea can be implemented in various ways. The algorithm implemented in our studies is as follows, see e.g. [12]:

Step 0 (Initialization) Set an initial temperature T and initial solution $\mathbf{x} = \mathbf{x}_{\text{curr}}$; determine the criterion value $F_C=F(\mathbf{x}_{\text{curr}})$
Step 1 Relative to the current value $\mathbf{x}_{\text{curr}}$, randomly determine a new value of $\mathbf{x}_{\text{new}} \in R^n$, and determine $F_N=F(\mathbf{x}_{\text{new}})$
Step 2 Let $d= F_N - F_C$. If $d < 0$ accept $\mathbf{x}_{\text{new}}$;, else, accept $\mathbf{x}_{\text{new}}$ only if a random variable U having the uniform p.d. on the interval [0,1] satisfies $U < \exp[-d/T]$. If $\mathbf{x}_{\text{new}}$ is accepted then $\mathbf{x}_{\text{curr}}$ is replaced by $\mathbf{x}_{\text{new}}$; else $\mathbf{x}_{\text{curr}}$ remains as is.
Step 3 Repeat steps 1 and 2 for given number K_T times.

Step 4 Lower T according to the annealing schedule and return to Step 1. Continue the process until the stopping criterion is met.
Step 5. Return the best solution $\boldsymbol{x_b}$ found during the cooling process and the value $-F(\boldsymbol{x_b})$

There are various specific implementations for the steps above in literature. In our simulation the initial solution $\mathbf{x} = \mathbf{x}_{curr}$ in Step 0 is set as the optimal solution of the DLP(Λ, β, χ) problem. The initial "temperature" T decays geometrically in the number of cooling phases (the number will be denoted as N_T in the sequel). Specifically, in the presented study the new temperature is related to the old temperature according to $T_{new} = 0.75\ T_{old}$. Another area for different implementations is in step 1, where $\mathbf{x}_{new}$ is generated randomly. In this study it was generated according the formula $\mathbf{x}_{new}=\mathbf{x}_{curr}+\boldsymbol{Z}$, where $\boldsymbol{Z}$ is a random n-dimensional vector with coefficients having expected value equal to zero and given standard deviation σ_z. Also note that the simulated annealing algorithm is design for minimization tasks, thus the criterion used in the algorithm is given by opposite values to the original ones which is reflected in the last step.

For any given problem it is likely that the performance of one algorithm will be superior to others. However, it is rarely possible to know a priori which algorithm is superior. Famous No Free Lunch theorems state that *when averaging over all optimization problems all search algorithms work the same (i.e., none can work better than a blind random search),* [12]. On the other hand the NFL theorems do not address the performance of a specific algorithm applied to a specific criterion function. They compare the performance of algorithms over all problems, where each problem is considered equally likely. It is well known that for some specific types of problems some algorithms may perform extremely well, while other perform very poorly.

Our aim is to determine which of the above algorithms (if any) is best for solving the stochastic linear programming problems.

5. SIMULATION STUDY OF THE ALGORITHMS

To compare the stochastic performance of various solutions $\mathbf{x}$ we use the performance indicators $P_f(\mathrm{x})$ and $SIP(\mathrm{x})$ given by (1) and (2), respectively. The values of the indicators $P_f(\mathrm{x})$ and $SIP(\mathrm{x})$ obviously depend on the problem parameters, i.e. on the tuple (Λ, β, χ) and on the dimensions of its elements. Thus in this study the values of the indicators are computed for various setups and then the statistical characteristics of the results are compared. Obviously, the value of the indicator $SIP(\mathrm{x})$ strongly depends on criterion function or, more precisely, on its maximum value achieved in the domain of feasible solutions. So, to make the comparisons more representative, it seems reasonable to compare in each case the value of the indicator $SIP()$ with the optimal value of the criterion function found in the corresponding deterministic problem DLP(Λ, β, χ). Consequently, in this study as an index of performance we use the following ratio

$$SDR(\mathbf{x}) = \frac{SIP(\mathbf{x})}{\max_D}$$

where $\max_D$ is the optimal objective function value found in corresponding deterministic case.

In order to obtain the statistical data containing the information about the performance of the examined algorithms we use the following simulation procedure. In this simulations a random CCP problem is generated N_P times and then its solutions and their characteristics are computed.

Step 0. Set the parameters n, K, N_{MC}, and K_T.
Step 1. Randomly generate the tuple (Λ, β, χ)
Step 2. Solve the DLP(Λ, β, χ) problem by the simplex algorithm and obtain the solution $\mathbf{x}_D$, the optimal value $\max_D = f(\mathbf{x}_D)$ and $SIP(\mathbf{x}_D)$
Step 3. Solve CCLP(**A,b,c**) problem with the help of a given GO algorithm, obtain the solution $\mathbf{x}$ and the related values of the criterion function (i.e. the indicators $P_f(\mathbf{x})$ or $SIP(\mathbf{x})$, respectively)
Step 4. Compute and record the values of the indices SDR for this setup
Step 5. Repeat the first to forth steps for N_P times, where N_P is a sufficiently large number.
Step 6. Return: statistical characteristics of the results such as maximum , minimum , mean values and standard deviations of $P_f()$ and SDR.

In our research we use the following values of the algorithms and simulation parameters: n=4,8,12 and K=K_T=10, N_P=50. In the ES-SS algorithm the distributions of mutations are normal with constant standard deviation equal to 0.1. The values of the parameter m (a number of constraints) are drawn from the set $\{n\text{-}2,...,n\text{+}5\}$. The elements of the tuple (Λ, β, χ) are drawn uniformly from the interval [-200,700].

One of the key issues in comparing algorithms is "efficiency" in solving problems of interest. The efficiency measure is some representation of the cost of finding an acceptable solution. There are many ways of measuring efficiency: computer run time, number of algorithm iterations, and number of the criterion function evaluations. As it was already pointed out in Introduction, in the considered approach the criterion function evaluation is realized via Monte Carlo simulations and thus it has the greatest impact on the total cost of the optimization process. In our simulations each criterion function value measurement requires N_{MC}=1000 realizations of the random problem setup . Thus, to make the comparison of the algorithms more objective, it was assumed that each GO algorithm is stopped after 300 evolutions of the criterion function. Then the performance indicators are computed. The results are presented in the next section.

6. SIMULATION RESULTS

In the following tables we present the results of the simulation comparison of the GO algorithms as tools for solving CCP problems.

Table 1 shows the statistical characteristics of the probability $P_f()$ computed for the best solution found with the help of the indicated algorithm as well as for the optimal solution found in the related deterministic problem (DP). The dimensions of the simulated problems are n=4,8,12.

Table 1 Statistical characteristics of the probability $P_f()$ computed for the best solutions

Algorithm	Min	Max	Mean	St.Dev.
	n=4			
ES-SS	0.672	1	0.992	0.046
GA	0.16	1	0.850	0.261
SA	0.048	1	0.908	0.263
DP	0.016	1	0.201	0.169
	n=8			
ES-SS	0.564	1	0.966	0.101
GA	0.076	1	0.847	0.288
SA	0.004	1	0.759	0.400
DP	0.004	0.452	0.089	0.096
	n=12			
ES-SS	0.136	1	0.872	0.211
GA	0.06	1	0.548	0.383
SA	0	1	0.597	0.441
DP	0	0.22	0.264	0.044

We see in Table 1 that all statistical characteristics of the results obtained for various dimensions of the problem show that the ES-SS algorithm outperforms the other methods with respect to this performance indicator. It is worth emphasising that the dominance of the ES-SS algorithm is indicated by all presented statistics and for all examined dimensions of the optimization problem So the ES-SS is the most effective algorithm in finding statistically feasible solutions (we remember that the parameter $P_f()$ indicates the probability of feasibility).

Next Table 2 presents analogous results, but this time related to the performance indicator *SDR*.

Table 2 Statistical characteristics of the indicator *SDR* computed for the best solutions

Algorithm	Min	Max	Mean	St.Dev.
	n=4			
ES-SS	0.381	1	0.759	0.148
GA	0.166	1	0.627	0.250
SA	0.005	0.932	0.698	0.233
DP	0.024	1	0.223	0.214
	n=8			
ES-SS	0.459	0.914	0.746	0.107
GA	0.074	0.946	0.595	0.242
SA	0.020	0.934	0.630	0.226
DP	0.004	0.252	0.082	0.069
	n=12			
ES-SS	0.105	0.887	0.660	0.163
GA	0.014	0.919	0.468	0.256

Algorithm	Min	Max	Mean	St.Dev.
	n=4			
ES-SS	0.381	1	0.759	0.148
GA	0.166	1	0.627	0.250
SA	0.005	0.932	0.698	0.233
DP	0.024	1	0.223	0.214
SA	0.004	0.907	0.438	0.315
DP	0	0.176	0.036	0.042

Similarly as in the previous table, results presented in Table 2 show that the ES-SS algorithm is better than the other considered methods. Although this time we see in the column containing the maximum values of *SDR*, that in some problems generated in our simulations other algorithms (GA or SA) perform better than ES-SS, but the average (mean) values of the performance indicator SDR as well as the remaining statistical characteristics describing the performance ofES-SS algorithm are evidently the best for all examined dimensions of the optimization problems.

7. FINAL REMARKS

Based on the presented results one can conclude that ES-SS algorithm is the most suitable tool for solving chance constrained programming problems, at least for problems that are similar to those considered in this paper. However, let us emphasize that though this algorithm demonstrates the best performance, the solution for CCLP(**A,b,c**) problem found with its help may be called satisfactory rather than optimal. The optimality cannot be proved, even more: we are aware that the solution is unlikely to be optimal. It is a typical situation when one use the global optimization algorithms based on stochastic search. On the other hand, taking into account the very complex nature of the optimization problem, the approach seems to be very attractive. The solution is relatively easy to find and, in considered stochastic framework, much better than the optimal solution for a corresponding deterministic problem. The solution found by ES-SS may also be considered as especially satisfactory because of the high values of the indicator *SDR*. Taking into account that the standard deviations of random variables disturbing all elements of the tuple (Λ, β, χ) amount to 10% of their original values, one should not expect much more, even when applying (in cases where it would be possible) more sophisticated mathematical tools.

REFERENCES

[1] CHARNES, A. – COOPER, W. W.: Chance-constrained programming, *Management Sciences* 6, pp. 73–80, 1959.

[2] DE, P. K. – ACHARYA, D. – SAHU K. C.: A chance-constrained goal programming model for capital budgetting, *Journal of the Operational Research Society,* Vol. 33, No. 7, pp. 635–638, 1982.

[3] GALAR, R.: *Soft selection in random global adaptation in Rn: A biocybernetic model of*

development, Technical University Press, Wrocław, 1990, (in Polish).

[4] KALL, P. – MAYER, J.: *Stochastic Linear Programming; Models, Theory, and Computatio*n, Springer, New York London, 2011.

[5] NORKIN, V. I. – PUG, G. C. – RUSZCZYŃSKI, A.: A branch and bound method for stochastic global optimization, *Mathematical Programming:* Series A and B, Vol. 83, No. 3, pp. 425–450, 1998.

[6] OBUCHOWICZ, A. – KORBICZ, J.: Global optimization via evolutionary search with soft selection, unpublished manuscript, available: http://www.mat.univie.ac.at/~neum/glopt/mss/gloptpapers.html

[7] RUSZCZYNSKI, A. – SHAPIRO, A.: *Stochastic Programming. Handbooks in Operations Research and Management Science*, Vol. 10. Elsevier, Amsterdam, 2003.

[8] SCHUËLLER, G. I. – JENSEN, H. A.: Computational methods in optimization considering uncertainties – An overview, *Comput. Methods Appl. Mech. Engrg*. Vol. 198, pp. 2–13, 2008.

[9] SEN, S. – HIGLE, J. L.: An introductory tutorial on stochastic linear programming models, *Interfaces*, Vol. 29, pp. 33–61, 1999.

[10] SEN, S.: Stochastic programming: computational issues and challenges, in: *Encyclopedia of Operations Research and Management Science, 2nd edition*, ed. S. Gass and C. Harris, Boston: Kluwer Academic Publishers, 2001, pp. 821–827.

[11] SLODIČÁK, V. – MACKO, P.: Some new approaches in functional programming using algebras and coalgebras, *Electronic Notes in Theoretical Computer Science* – ENTCS 279, Elsevier, 2011, pp. 41–62.

[12] SPALL, J. C.: *Introduction To Stochastic Search And Optimization; Estimation, Simulation, And Control*, A John Wiley & Sons. Inc., Publication, 2003.

[13] STEUER, R. – NA, P.: Multiple criteria decision making combined with finance: A categorized bibliographic study, *European Journal of Operational Research,* Vol. 150, pp. 496–515, 2003.

[14] WEISE, T.: Global *Optimization Algorithms – Theory and Application*, available http://www.it-weise.de/projects/book.pdf (accessed 7.09.2011)

Received October 12, 2011, accepted December 30, 2011

BIOGRAPHY

Andrzej Z. Grzybowski was born on 29.05.1958. In 1982 he graduated (MSc in mathematics) at the department of Fundamental Problems of Technology at the Wroclaw University of Technology. He defended his PhD in the field of mathematics in 1991; his thesis title was "Minimax decisions in some problems of control of discrete-time stochastic systems". Since 1983 he works at the Czestochowa University of Technology. Since 1991 he is a Head of the Applied Mathematics Subunit at Institute of Mathematics at the Department of Mechanical Engineering and Computer Science. His scientific research is focusing on formal and Monte Carlo analysis of various problems arising in statistical decision theory.