Evolutionary Optimization for Decision Making under Uncertainty

Ronald Hochreiter

May 2011

Abstract

Optimizing decision problems under uncertainty can be done using a variety of solution methods. Soft computing and heuristic approaches tend to be powerful for solving such problems. In this overview article, we survey Evolutionary Optimization techniques to solve Stochastic Programming problems - both for the single-stage and multi-stage case.

Keywords: Optimization under Uncertainty, Optimal Decision Making, Evolutionary Optimization, Stochastic Programming, Multi-stage Stochastic Programming.

1 Introduction

There are many approaches towards solving decision problems under uncertainty. Throughout this paper, we will focus on solving Stochastic Programming problems - both the classical single-stage as well as the multi-stage case. This article intends to provide ideas on how to use Evolutionary Optimization techniques to solve this class of optimization problems.

To summarize the concept of Stochastic Programming, we will refer to the summary of [10]: consider a deterministic decision optimization problem, where a decision maker aims at finding an optimal (numerical) decision $x \in \mathbb{R}^n$ by minimizing a deterministic cost function $f(\cdot)$ (or by maximizing a profit function respectively) given a set $\mathcal{X}$ of constraints, which generally consists of various physical, organizational, and regulatory restrictions. The mathematical formulation of this problem can be simplified to the formulation shown in Equ. (1).

$$\begin{array}{ll} \text{optimize } x: & f(x) \\ \text{subject to} & x \in \mathcal{X}. \end{array} \tag{1}$$

During the 1950s Stochastic Programming was initiated by Dantzig [5] and Beale [2]. The idea is to replace deterministic parameters by probability distributions on some probability space $(\Omega, \mathcal{F}, \mathbb{P})$, which will be denoted by Ξ in the following, and to optimize a stochastic cost (or profit) function $f(\cdot, \cdot)$ over some probability functional $\mathbb{F}$. A common choice regarding this functional is the expectation $\mathbb{E}$. As Rockafellar [25] points out, expectations are only suitable for situations where the interest lies in long-range operation, and stochastic ups and downs can safely average out, which is not the case e.g. for managing financial risks. The recent progress of unifying probabilistic risk measures, as

presented in the seminal paper by Artzner et al. [1] on coherent risk measures, motivated for using probability functionals based on risk measures. See the book [23] for more details on modeling, measuring, and managing risk for this class of optimization applications. A different view on integration of risk measures using the concept of deviation measures is shown in [26]. In summary, the resulting mathematical meta-formulation of a stochastic program for arbitrary probability functionals is shown in Equ. (2).

$$\begin{array}{ll} \text{optimize } x: & \mathbb{F}(f(x,\Xi)) \\ \text{subject to} & (x,\Xi) \in \mathcal{X}. \end{array} \tag{2}$$

However, a concrete reformulation of this meta-model into some model, which can be solved with a numerical optimizer depends to a high degree on the chosen probability functional, as well as on the structure of the underlying probability space.

The interested reader is referred to [27] for a theoretical overview of the area of Stochastic Programming, and to [35] for Stochastic Programming languages, environments, and applications. Interestingly, evolutionary approaches have not been applied to a wide range of real stochastic programming problems so far, only scarce examples are available, e.g. recent works in the field of chemical batch processing, see [34] and [31].

This paper is organized as follows. Section 2 briefly describes the heuristic (soft computing) approach, which was chosen to solve the stochastic programs. Section 3 and Section 4 consider the single-stage and the multi-stage case respectively. In both Sections the optimization as well as the scenario generation aspect will be discussed. Section 5 concludes the paper.

2 Evolutionary stochastic optimization

All stochastic optimization problems throughout this paper will be solved by adapting a standard Evolutionary Optimization algorithm, e.g. as surveyed by [3] and summarized below, to handle the stochastic meta-model directly without complex reformulations.

$P \leftarrow GenerateInitialPopulation$
$Evaluate(P)$
while termination conditions not met **do**
  $P' \leftarrow Recombine(P)$
  $P'' \leftarrow Mutate(P')$
  $Evaluate(P'')$
  $P \leftarrow Select(P \cup P'')$
end while

As mentioned above, this meta-formulation of Evolutionary Optimization can be applied to the meta-formulation of Stochastic Programming shown in Equ. (2) in different ways. We consider the case of finite, discrete scenario sets in this paper. This is not a serious restriction as many real-world problems can be satisfyingly described with such sets. In addition, analytical solutions and continuous descriptions of the uncertainty often imply a gross underestimation of the complexity of the respective problem.

3 Single-stage stochastic optimization

In single-stage stochastic optimization, the input is given by a multi-variate probability distribution - a discrete one in our special application-related approach. In the following, issues raised by using Evolutionary Optimization techniques are shown using the well-known application of financial risk-return portfolio selection. In a second step, scenario generation for the single-stage case is boiled down to being solvable using an evolutionary approach.

3.1 Single-stage optimization: Portfolio selection

An ideal application example is financial risk-return portfolio optimization. For various reasons, this optimization problem naturally fits into the general problem structure usually handled with evolutionary techniques. Therefore, evolutionary approaches have been successfully applied to different classes of portfolio optimization problems, see e.g. [29], [28], [30], [8], [17], as well as the references therein, or refer to [18]. Some of the approaches mention the field of Stochastic Programming explicitly, e.g. [32], and [36].

The classical bi-criteria portfolio optimization problem based on Markowitz [19] can be summarized as follows: An investor has to choose a portfolio from a set of (financial) assets $\mathcal{A}$ with finite cardinality $a = |\mathcal{A}|$ to invest her available budget. The bi-criteria problem stems from the fact that the investor aims at maximizing her return while aiming at minimizing the risk of the chosen portfolio at the same time.

In the single-stage stochastic programming case, the underlying uncertainty is represented as a multi-variate probability distribution on the respective probability space. The distribution may either be continuous or discrete. The discretized probability distribution, which is used to compute the optimal decision is called *scenario set*. This discrete set of scenarios $\mathcal{S}$ has finite cardinality $s = |\mathcal{S}|$, where each s_i is equipped with a non-negative probability $p_i \geq 0$, and $\sum_{j=1}^{s} p_j = 1$. From the financial market viewpoint, each scenario contains one possible set of joint future returns of all a assets under consideration for the portfolio. Using the terminology of Markowitz, each scenario contains the discounted anticipated return of each asset.

Let $x \in \mathbb{R}^a$ be some portfolio. Without loss of generality, we use budget normalization, i.e. $\sum_{a \in \mathcal{A}} x_a = 1$. Each component x_i of the portfolio vector denotes the fraction of the available budget B invested into the respective asset i. We may now rewrite the scenario set $\mathcal{S}$ as a matrix S to calculate the discrete Profit & Loss (P&L) distribution ℓ for some portfolio x, which is simply the cross product $\ell_x = \langle x, S \rangle$. We will denote ℓ the loss distribution in the following. Finally, let $x^*_\rho \in \mathbb{R}^a$ denote the optimal portfolio given some risk measure ρ and ℓ^*_ρ denote the respective ρ-optimal discrete loss distribution.

While this model clearly represents a multi-criteria optimization model, we apply a criteria-weighted model, where an additional risk-aversion parameter κ is defined, i.e.

$$\begin{array}{ll} \text{maximize } x: & \mathbb{E}(\ell_x) - \kappa\rho(\ell_x), \\ \text{subject to} & x \in \mathcal{X}. \end{array} \tag{3}$$

An important fact is that this problem already shows the issue that one really has to care about the most is the genotype-phenotype encoding. The most straightforward approach to design a usable genotype-phenotype representation

of the portfolio selection problem is to use a vector of real values between 0 and 1 and normalize the resulting vector to the sum of the available budget, i.e. in the most simple case to a budget of 1. However, it can be shown that this approach rarely leads to stable solutions, i.e. two runs almost never converge to the same portfolio composition, see also [12].

Hence, we need a special genetic encoding of a portfolio, see [29]. Each gene thereby consists of two parts: One that determines the amount of budget to be distributed to each selected asset and one part which determines in which assets to invest. The first part g_1 consists of a predefined number b of real values between 0 and 1 and the second part g_2 is encoded as a bit-string of the size of the amount of assets.

Example (from [12]). Let us define a bucket size of $b = 10$ and consider that want to select the optimal portfolio out of $a = 5$ assets. A random chromosome with a fixed number of 3 asset picks may then consist of the following two parts g_1 and g_2.

$$g_1 = (0.4893, 0.3377, 0.9001, 0.3692, 0.1112, 0.7803, 0.3897, 0.2417, 0.4039, 0.0965)$$

$$g_2 = (1, 1, 0, 0, 1)$$

If we re-map these two parts, we receive the following portfolio x

$$x = (0.3, 0.5, 0, 0, 0.2),$$

which is a valid portfolio composition, and can be used to calculate the loss function and to conduct a full evaluation of the respective portfolio optimization formulation within the evolutionary optimization process.

With this Evolutionary Optimization strategy a full-fledged portfolio optimization tool for a plethora of risk measures can be created, as the evolutionary approach only needs to evaluate the risk and not directly optimize over the complete (probably non-convex) problem. Furthermore, all kinds of constraints, even non-convex ones, may be included. Constraint handling can be easily added using a penalty structure within the objective function evaluation.

3.2 Single-stage scenario generation

The problem of single-stage scenario generation, i.e. an optimal approximation of a multi-variate probability distribution can be done via various sampling as well as clustering techniques. Optimal single-stage approximation have been done by using e.g. Principal Component Analysis (see e.g. [33]), K-Means clustering (see e.g. [14]), or moment matching ([15]). The result is directly affected by the chosen approximation method - both a distance and a heuristic needs to be specified.

Of course, this problem can be solved with Evolutionary Optimization techniques as well. Consider the following example (from [13]) of a full-fledged scenario generation procedure: let's assume that we do have 10 input scenarios (asset returns), each equipped with the same probability $p = 0.1$, which might be the output of some sophisticated asset price sampling procedure, e.g.

$$(0.017, -0.023, -0.008, -0.022, -0.019, 0.024, 0.016, -0.006, 0.032, -0.023).$$

We want to separate those values optimally into 2 clusters, which then represent our output scenarios and take a random chromosome, which might look as follows:

$$(0.4387, 0.3816, 0.7655, 0.7952, 0.1869, 0.4898, 0.4456, 0.6463, 0.7094, 0.7547)$$

If we map this vector to represent 2 centers we obtain: $(1, 1, 2, 2, 1, 1, 1, 2, 2, 2)$. Now we need to calculate a center value, e.g. the mean, and have to calculate the distance for each value of each cluster to its center, e.g. we obtain center means $(0.0032, -0.0055)$, which represent the resulting scenarios, each with a probability of 0.5. The l_1 distance for each cluster is $(0.0975, 0.0750)$, so the objective function value is 0.1725. Now flip-mutate chromosome 9, i.e. $(1 - 0.7094) = 0.2906$, such that input scenario 9 (return = 0.032) will now be part of cluster 1 instead of cluster 2. We obtain new scenarios $(0.0080, -0.0149)$ with probabilities $(0.6, 0.4)$. The objective function value is 0.1475 (or 0.1646 if you weight the distances with the corresponding output scenario probability), i.e. this mutation led to a better objective value.

3.3 Wrapping up the single-stage case

In this Section we have shown that Evolutionary Optimization can be conveniently be applied to single-stage stochastic optimization problems - both to the solution of decision models as well as to the creation of scenario sets, i.e. for the process of scenario generation. However, one has to be careful with choosing an appropriate genotype-phenotype representation, as the most straightforward approach might not lead to a stable solution.

4 Multi-stage stochastic optimization

To summarize the case of multi-stage stochastic optimization, we base the following summary on [11]: we consider algorithmic issues of the computation of multi-stage scenario-based stochastic decision processes for decision optimization models under uncertainty. We consider that given a multi-stage stochastic programming problem specific discrete-time stochastic process on the decision horizon $t = 1, \ldots, T$, a decision maker observes the realization of this random process ξ_t at each decision stage t, and takes a decision x_t based on all observed values up to t $(\xi_1, \ldots, \xi_t)$. Let there now be a sequence of decisions $x_1, \ldots, x_T$. At the terminal stage T we observe a sequence of decisions $x = (x_1, \ldots, x_T)$ with realizations $\xi = (\xi_1, \ldots, \xi_T)$, which lead to cost $f(x, \xi)$ (or likewise profit). The stochastic optimization task is to find the sequence of decisions $x(\xi)$, which minimizes some probability functional (most commonly the expectation, a risk measure, or a combination of these two) of the respective cost function $f(x(\xi), \xi)$ - see e.g. [7] for a classification of risk measures in this context.

We consider the multi-stage case in such a way that there is at least one intermediary stage between root and terminal stage, i.e. $T > 2$. Unfortunately, the topic of scenario generation is not sufficiently treated in most text books on stochastic programming. Let us consider multi-stage stochastic programming

problems as defined in Equ. (4).

$$\begin{array}{ll} \text{minimize } x: & \mathbb{F}\big(f(x(\xi),\xi)\big) \\ \text{subject to} & (x(\xi),\xi) \in \mathcal{X} \\ & x \in \mathcal{N} \end{array} \tag{4}$$

The multi-variate, multi-stage stochastic process ξ describes the future uncertainty, i.e. the subjective part of the stochastic program, and the constraint set $\mathcal{X}$ defines feasible combinations of x and ξ. This constraint set is used to model the underlying real-world decision problem. Furthermore, a set of non-anticipativity constraints $\mathcal{N}$, consisting of functions $\xi \mapsto x$ which make sure that x_t is only based on realizations up to stage t $(\xi_1, \ldots, \xi_t)$, is necessary. To solve multi-stage programs with numerical optimization solvers, the underlying stochastic process has to be discretized into a scenario tree, and this scenario tree approximation will inherently fulfill these non-anticipativity constraints.

Two issues negatively affect the application of multi-stage stochastic programs for real-world decision problems: First, modeling the underlying decision problem is a non-trivial task. Multi-stage optimization models and stochastic scenario models require stable scenario tree handling procedures, which are considered to be too cumbersome to be applied to real-world applications, and the communication of tree-based models to non-experts is complicated. Secondly, modeling the underlying uncertainty is complex and messy. As discussed above, a good discrete-time, discrete-space scenario tree approximation of the underlying stochastic process has to be generated in order to numerically compute a solution and the quality of the scenario model severely affects the quality of the solution.

4.1 Solving the scenario tree optimization problem

The quality of the scenario tree severely affects the quality of the solution of the multi-stage stochastic decision model, such that any approximation should be done in consideration of optimality criteria, i.e. before a stochastic optimization model is solved, a scenario optimization problem has to be solved independently of the optimization model. It should be noted, that there are also scenario generation approaches where the generation is not decoupled from the optimization procedure, see e.g. [4]. A major drawback is that these methods are often limited to a certain restricted set of models, and cannot be generalized easily.

In the context of separated scenario optimization, optimality can be defined as the minimization of the distance between the original (continuous or highly discrete) stochastic process and the approximated scenario tree. Choosing an appropriate distance may be based on subjective taste, e.g. Moment Matching as proposed by [15], selected due to theoretical stability considerations (see [24] and [9]), which leads to probability metric minimization problems as shown by [22] and [6], or it may be predetermined by chosen approximation method, e.g. by using different sampling schemes like QMC in [21] or RQMC in [16], see also [20]. It is important to remark that once the appropriate distance has been selected, an appropriate heuristic to approximate the chosen distance has to be applied, which affects the result significantly. The real algorithmic challenge of multi-stage scenario generation is caring about the tree structure while still minimizing the distance. Only in rare cases, this problem can be solved without heuristics.

The following evolutionary approach has been presented by [13]. We assume that there is a finite set $\mathcal{S}$ of multi-stage, multi-variate scenario paths, which are sampled using the preferred scenario sampling engine selected by the decision taker. Stages will be denoted by $t = 1, \ldots, T$ where $t = 1$ represents the (deterministic) root stage (root node), and T denotes the terminal stage. Therefore, the input consists of a scenario path matrix of size $|\mathcal{S}| \times (T-1)$. Furthermore, the desired number of nodes of the tree in each stage is required, i.e. a vector n of size $(T-1)$.

We will focus on the uni-variate case. However, the extension to the multi-variate case does not pose any structural difficulties besides that a dimension-weighting function for calculating the total distance on which the optimality of the scenario tree approximation is based on has to be defined.

A crucial part in designing a multi-stage scenario tree generator based on evolutionary techniques is finding a scalable genotype representation of a tree, as this is always the case for Evolutionary Optimization approaches - specifically both in terms of the numbers of stages as well as the number of input scenarios. The approach surveyed here is using a real-valued vector in the range $[0, 1]$ and mapping it to a scenario tree given the respective node format n. The length of the vector is equal to the number of input scenarios $s = |\mathcal{S}|$ plus the number of terminal nodes n_T. Thus, the presented algorithm is somewhat limited by the number of input scenarios. This means that input scenarios should not simply be a standard set of mindlessly sampled scenario paths, but rather a thoughtfully simulated view on the future uncertainty. This should not be seen as a drawback, as it draws attention to this often neglected part of the decision optimization process.

To map the real-valued vector to a scenario tree, which can be used for a subsequent stochastic optimization, two steps have to be fulfilled. First, the real-valued numbers are mapped to their respective node-set given the structure n of the tree, and secondly, values have to be assigned to the nodes. It should be noted, that a random chromosome does not necessarily lead to a valid tree. This is the case if the number of mapped nodes is lower than the number of nodes necessary given by n_t of the respective stage t. If an uniform random variable generator and a thoughtful node structure is used, which depends on the number of input scenarios, invalid trees should not appear frequently, and can be easily discarded if they do appear during the Evolutionary Optimization process.

We may now conveniently use the Evolutionary Optimization clustering strategy, which has been shown for creating single-stage scenarios above. This approach seems to be rather trivial for the single-stage case, but this simple approach leads to a powerful method for the tedious task of constructing multi-stage scenario trees for stochastic programming problems, because the nested probability structure of the stochastic process is implicitly generated.

For multi-stage trees, a crucial point is finding a representative value for the node-sets determined in the first step of the mapping. There exists a range of methods, which can be used for the determination of centers, i.e. mapping all values of a node-sets to one node. The distance of the approximation, which is used for the calculation of the objective function, will also be affected by this method. A straight-forward solution is to use the median of the values - see [13] for other methods and their differences. Fig. 1 shows an example of the algorithm, i.e. 200 sampled paths are used to create a multi-stage $[1, 10, 40]$ scenario

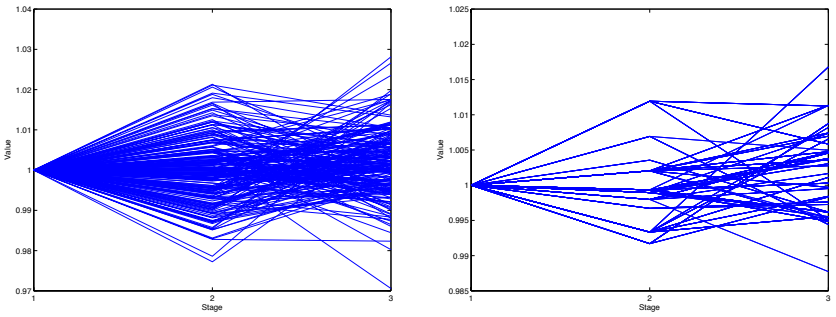

Figure 1: Input scenarios ($n = 200$) (left) and a scenario tree with $n = [10, 40]$ (right).

tree. In summary, the complex task of generating multi-stage scenario trees can be handled conveniently by using an Evolutionary Optimization approach.

4.2 Solving the multi-stage decision problem

Solving a multi-stage stochastic programming problem with Evolutionary Optimization techniques is not straightforward. Especially when the underlying scenario tree is huge, because one would have to calculate an optimal decision on every node, such that the chromosome would simply be too large, i.e. for each node of the tree the respective amount of genes (related to the multi-variate structure of the decision problem) would have to be added to the chromosome. New simplification strategies have to be devised, which are out of the scope of this paper.

5 Conclusion

In this paper, several methods on how to apply Evolutionary Optimization methods to solve stochastic programs have been surveyed and outlined. It is shown that this class of heuristic optimization solvers can be applied to these problems conveniently. However, one has to take care of a few pitfalls during the creation of specific solution methods, which were discussed in this paper.

Acknowledgement: The author is grateful to the MENDEL conference organizing committee for being invited to present this research to the soft computing community.

References

[1] P. Artzner, F. Delbaen, J-M. Eber, and D. Heath. Coherent measures of risk. *Mathematical Finance*, 9(3):203–228, 1999.

[2] E. M. L. Beale. On Minimizing a Convex Function Subject to Linear Inequalities. *Journal of the Royal Statistical Society, Series B*, 17:173–184, 1955.

[3] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3):268–308, 2003.

[4] M. S. Casey and S. Sen. The scenario generation algorithm for multistage stochastic linear programming. *Mathematics of Operations Research*, 30(3):615–631, 2005.

[5] G. B. Dantzig. Linear Programming under Uncertainty. *Management Science*, 1:197–206, 1955.

[6] J. Dupačová, N. Gröwe-Kuska, and W. Römisch. Scenario reduction in stochastic programming. An approach using probability metrics. *Mathematical Programming*, 95(3, Ser. A):493–511, 2003.

[7] A. Eichhorn and W. Römisch. Polyhedral risk measures in stochastic programming. *SIAM Journal on Optimization*, 16(1):69–95, 2005.

[8] M. A. Gomez, C. X. Flores, and M. A. Osorio. Hybrid search for cardinality constrained portfolio optimization. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1865–1866. ACM Press, 2006.

[9] H. Heitsch, W. Römisch, and C. Strugarek. Stability of multistage stochastic programs. *SIAM Journal on Optimization*, 17(2):511–525, 2006.

[10] R. Hochreiter. Evolutionary stochastic portfolio optimization. In A. Brabazon and M. O'Neill, editors, *Natural Computing in Computational Finance*, volume 100 of *Studies in Computational Intelligence*, pages 67–87. Springer, 2008.

[11] R. Hochreiter. Algorithmic aspects of scenario-based multi-stage decision process optimization. In F. Rossi and A. Tsoukiàs, editors, *Algorithmic Decision Theory 2009*, volume 5783 of *Lecture Notes in Computer Science*, pages 365–376. Springer, 2009.

[12] R. Hochreiter. An algorithm for evolutionary stochastic portfolio optimization with probabilistic constraints. In R. Matousek, editor, *16th International Conference on Soft Computing (MENDEL 2010)*, volume 16 of *Mendel Conference*, pages 1–6. VUT Press, 2010.

[13] R. Hochreiter. Evolutionary multi-stage financial scenario tree generation. In C. Di Chio et al., editor, *Applications of Evolutionary Computation, Part II*, volume 6025 of *Lecture Notes in Computer Science*, pages 182–191. Springer, 2010.

[14] R. Hochreiter and G. Ch. Pflug. Financial scenario generation for stochastic multi-stage decision processes as facility location problems. *Annals of Operations Research*, 152(1):257–272, 2007.

[15] K. Høyland and S. W. Wallace. Generating scenario trees for multistage decision problems. *Management Science*, 47(2):295–307, 2001.

[16] M. Koivu. Variance reduction in sample approximations of stochastic programs. *Mathematical Programming*, 103(3, Ser. A):463–485, 2005.

[17] D. Lin, X. Li, and M. Li. A genetic algorithm for solving portfolio optimization problems with transaction costs and minimum transaction lots. In L. Wang, K. Chen, and Y.-S. Ong, editors, *Advances in Natural Computation, First International Conference, ICNC 2005, Changsha, China, August 27-29, 2005, Proceedings, Part III*, volume 3612 of *Lecture Notes in Computer Science*, pages 808–811. Springer, 2005.

[18] D. Maringer. *Portfolio Management with Heuristic Optimization*, volume 8 of *Advances in Computational Management Science*. Springer, 2005.

[19] H. M. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

[20] T. Pennanen. Epi-convergent discretizations of multistage stochastic programs via integration quadratures. *Mathematical Programming*, 116(1-2, Ser. B):461–479, 2009.

[21] T. Pennanen and M. Koivu. Epi-convergent discretizations of stochastic programs via integration quadratures. *Numerische Mathematik*, 100(1):141–163, 2005.

[22] G. Ch. Pflug. Scenario tree generation for multiperiod financial optimization by optimal discretization. *Mathematical Programming*, 89(2, Ser. B):251–271, 2001.

[23] G. Ch. Pflug and W. Römisch. *Modeling, Measuring and Managing Risk*. World Scientific, 2007.

[24] S. T. Rachev and W. Römisch. Quantitative stability in stochastic programming: the method of probability metrics. *Mathematics of Operations Research*, 27(4):792–818, 2002.

[25] R. T. Rockafellar. Coherent approaches to risk in optimization under uncertainty. Tutorial Notes, Version May 14th, 2007, 2007.

[26] R. T. Rockafellar, S. Uryasev, and M. Zabarankin. Generalized deviations in risk analysis. *Finance and Stochastics*, 10(1):51–74, 2006.

[27] A. Ruszczyński and A. Shapiro, editors. *Stochastic programming*, volume 10 of *Handbooks in Operations Research and Management Science*. Elsevier Science B.V., Amsterdam, 2003.

[28] F. Schlottmann, A. Mitschele, and D. Seese. A multi-objective approach to integrated risk management. In C. A. Coello Coello, A. H. Aguirre, and E.Zitzler, editors, *Proceedings of the Evolutionary Multi-Criterion Optimization Conference (EMO 2005)*, volume 3410 of *Lecture Notes in Computer Science*, pages 692–706. Springer, 2005.

[29] F. Streichert, H. Ulmer, and A. Zell. Evolutionary algorithms and the cardinality constrained portfolio selection problem. In *Selected Papers of the International Conference on Operations Research (OR 2003)*, pages 253–260. Springer, 2003.

[30] R. Subbu, P.P. Bonissone, N. Eklund, S. Bollapragada, and K. Chalermkraivuth. Multiobjective financial portfolio design: a hybrid evolutionary approach. In *The 2005 IEEE Congress on Evolutionary Computation*, volume 2, pages 1722–1729. IEEE Press, 2005.

[31] J. Till, G. Sand, M. Urselmann, and S. Engell. A hybrid evolutionary algorithm for solving two-stage stochastic integer programs in chemical batch scheduling. *Computers & Chemical Engineering*, 31(5-6):630–647, 2007.

[32] K.-i. Tokoro. A statistical selection mechanism of ga for stochastic programming problems. In *Proceedings of the 2001 Congress on Evolutionary Computation*, volume 1, pages 487–492, 2001.

[33] N. Topaloglou, H. Vladimirou, and S.A. Zenios. Cvar models with selective hedging for international asset allocation. *Journal of Banking and Finance*, 26(7):1535–1561, 2002.

[34] M. Urselmann, M.T.M. Emmerich, J. Till, G. Sand, and S. Engell. Design of problem-specific evolutionary algorithm/mixed-integer programming hybrids: Two-stage stochastic integer programming applied to chemical batch scheduling. *Engineering Optimization*, 39(5):529–549, 2007.

[35] S. W. Wallace and W. T. Ziemba, editors. *Applications of stochastic programming*, volume 5 of *MPS/SIAM Series on Optimization*. Society for Industrial and Applied Mathematics (SIAM), 2005.

[36] S.-M. Wang, J.-C. Chen, H.-M. Wee, and K.-J. Wang. Non-linear stochastic optimization using genetic algorithm for portfolio selection. *International Journal of Operations Research*, 3(1):16–22, 2006.