Gradient Based Stochastic Mutation Operators in Evolutionary Multi-objective Optimization

Pradyumn Kumar Shukla

Institute of Numerical Mathematics, Department of Mathematics,
Dresden University of Technology, Dresden PIN 01069, Germany
pradyumn.shukla@mailbox.tu-dresden.de

Abstract. Evolutionary algorithms have been adequately applied in solving single and multi-objective optimization problems. In the single-objective case various studies have shown the usefulness of combining gradient based classical search principles with evolutionary algorithms. However there seems to be a dearth of such studies for the multi-objective case. In this paper, we take two classical search operators and discuss their use as a local search operator in a state-of-the-art evolutionary algorithm. These operators require gradient information which is obtained using a stochastic perturbation technique requiring only two function evaluations. Computational studies on a number of test problems of varying complexity demonstrate the efficiency of hybrid algorithms in solving a large class of complex multi-objective optimization problems.

1 Introduction

Multi-objective optimization is a rapidly growing area of research and application in modern-day optimization. There exist a plethora of non-classical methods which follow some natural or physical principles for solving multi-objective optimization problems, see for example the book by [4]. On the other hand a large amount of studies have been devoted to develop classical methods for solving multi-objective optimization problems.

Evolutionary algorithms use stochastic transition rules using crossover and mutation search operators to move from one solution to another. In this way global structure of search space is exploited. Classical methods, on the other hand, usually use deterministic (usually gradient based) transition rules to move from one solution to another. Classical methods effectively use local information thus ensuring fast convergence. This however comes up at the cost of requiring gradient or Hessian information which requires a large number of function evaluations. Hence one sees that there is a trade-off between fast convergence and number of function evaluations. Hybrid implementations thus continue to be developed and tested (see for example [1,8,9,2,3]).

This study is motivated by the an earlier comparative study [14]. In this contribution we take two classical gradient based Pareto front generating methods and use their search principles as mutation operators in a state-of-the-art multi-objective evolutionary algorithm to create a powerful hybrid multi-objective metaheuristics algorithm. We demonstrate their efficiency in solving real valued problems of varying complexity.

This paper is structured as follows. The next section present an overview of various classical generating methods and the gradient estimation technique, while the third section presents the simulation results. Conclusions as well as extensions which emanated from this study are presented in the end of this contribution.

2 Classical Search Principles

For the present study we take two classical algorithms and use their search operators as mutation operators in the elitist non-dominated sorting GA or NSGA-II developed by [5]. The gradients of objective functions are obtained by a stochastic method described later in this section. These classical algorithms and their search operators are described next.

2.1 Schäffler's Stochastic Method (SSM)

This method [13], is based on the solution of a set of stochastic differential equations. It method requires the objective functions to be twice continuously-differentiable. In each iteration, a trace of non-dominated points is constructed by calculating at each point $\mathbf{x}$ a direction $(-q(\mathbf{x}))$ in the decision space which is a direction of descent for *all* objective functions (note that we consider m to be the number of objective functions denoted by f_i for all $i = 1, 2, \cdots, m$). The direction of descent is obtained by solving a quadratic subproblem. Let $\hat{\alpha}$ be the minimizer. Then $q(\mathbf{x}) = \sum_{i=1}^{m} \hat{\alpha}_i \nabla f_i(\mathbf{x})$. A set of non-dominated solution is obtained by perturbing the solution (minimum along the direction of descent) using a Brownian motion concept. The following stochastic differential equation (SDE) is employed for this purpose:

$$d\mathbf{X}_t = -q(\mathbf{X}_t)d(t) + \varepsilon dB_t, \quad \mathbf{X}_0 = \mathbf{x}_0, \tag{1}$$

where $\varepsilon > 0$ and B_t is a n-dimensional Brownian motion. As the first search operator we use Equation 1 to create a child instead of the mutation operator. The gradients are obtained using a stochastic perturbation method described later. We name the hybrid algorithm with new mutation operator as S-NSGA-II. In all simulations here, to solve the above equation numerically, we employ the Euler's method with a step size σ. The approach needs two parameters to be set properly: (i) the parameter ε which controls the amount of global search and (ii) the step size σ used in the Euler's approach which controls the accuracy of the integration procedure.

2.2 Timmel's Population Based Method (TPM)

As early as in 1980, [16,17] proposed a population-based stochastic approach for finding multiple Pareto-optimal solutions of a differentiable multi-objective optimization problem. In this method, first, a feasible solution set (we call it a population) is randomly created. The non-dominated solutions ($\mathbf{X}_0=\{\mathbf{x}_1^0, \mathbf{x}_2^0, \ldots, \mathbf{x}_s^0\}$) are identified and they serve as the first approximation to the Pareto-optimal set. Thereafter, from each solution $\mathbf{x}_k^0$, a child solution is created in the following manner:

$$\mathbf{x}_k^1 = -\sum_{i=1}^{M} t_1 u_i \nabla f_i(\mathbf{x}_k^0), \tag{2}$$

where u_i is the realization of a uniformly distributed random number (between 0 and 1) and t_1 is step-length in the first generation. It is a simple exercise to show that the above formulation ensures that not all functions can be worsened simultaneously. The variation of the step-length over iterations must be made carefully to ensure convergence to the efficient frontier. The original study suggested the following strategy for varying the step-length t_j with generation j: $t_j = C/j$ (where C is a positive constant). As the second search operator we use Equation 2 to create a child instead of the mutation operator. As in S-NSGA-II the gradients are obtained using a stochastic perturbation method described later. We name the hybrid algorithm with new mutation operator as T-NSGA-II.

2.3 Gradient Estimation: Simultaneous Perturbation Method

In almost all classical algorithms (for both single and multi-objective problems) the gradient of a function (say in general h) are required. The standard approach for estimating the gradient is the Finite Difference (FD) method. For variable (say $\mathbf{x}$) of dimension n this method of gradient estimation requires $2n$ function evaluations. This is costly in terms of function evaluations (of the order $O(n)$). The Simultaneous Perturbation (SP) method [15] on the other hand requires only *two* function evaluation independently of n (computational complexity is thus $O(1)$) as follows

$$g_i(\mathbf{x}) = \frac{f(x + c\mathit{\Delta}) - f(x - c\mathit{\Delta})}{2c\mathit{\Delta}_i},$$

where the i^{th} component of the gradient is denoted by $g_i(\mathbf{x})$, $\mathit{\Delta}$ is a n dimensional vector of random perturbations satisfying certain statistical conditions ([15]). A simple (and theoretically valid) choice for each component of $\mathit{\Delta}$ is to use a Bernoulli distribution ± 1 with probability of 0.5 for each ± 1 outcome. The step size c at each iteration (denoted by c_k) is given as $c_k = c_0/(k+1)^{\gamma}$. Practically effective (and theoretically valid [15]) values of c_0, γ are 0.001 and 1/6 which are used here.

3 Simulation Results

In this section, we compare the above two hybrid methods with the elitist non-dominated sorting GA or NSGA-II [5] on a number of test problems. The test problems are chosen in such a way so as to systematically investigate various aspects of an algorithm. For S-NSGA-II the parameters $\sigma = 1.0$ along with $\epsilon = 0.1$ is used for all the test problems. For T-NSGA-II the parameter $C = 10.0$ is used (unless otherwise stated). For the NSGA-II, we use a standard real-parameter SBX and polynomial mutation operator with $\eta_c = 10$ and $\eta_m = 10$, respectively [4] (unless otherwise stated). For all problems solved, we use a population of size 100. We set the number of function evaluations as 5000 for each problems and 15000 for difficult ones.

Convergence and diversity are two distinct goals in multi-objective optimization. In order to evaluate convergence we use the Generational Distance (GD) metric [4]. Diversity of solutions is evaluated using the Spread (denoted by S) metric [4]. These unary metrices for convergence and diversity are used together with a binary metric which can detect whether an approximation set is better than another. We use the multiplicative binary ϵ indicator discussed by Zitzler [18] to assess the performance of the algorithms. Also for statistical evaluation we use attaintment surface based statistical metric ([7]) for one hard problem. We consider two-objective ZDT test problems discussed in [4]. The test problems are slightly modified so that they become unconstrained multi-objective optimization problems, as the SSM method is only able to tackle unconstrained problems in its present form. Table 1 present these test problems. Also the box constraints are slightly modified so that the functions are twice continuously differentiable in the entire feasible region (required as per SSM).

Table 1. Test problems

Name	Objectives	g	Domain
ZDT1	$f_1(\mathbf{x}) = x_1, f_2(\mathbf{x}) = g(\mathbf{x})\left(2 - \sqrt{\frac{x_1}{g(\mathbf{X})}}\right)$	$g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i^2$	$[0.01, 1]\times[-1, 1]^{29}$
ZDT2	$f_1(\mathbf{x}) = x_1, f_2(\mathbf{x}) = g(\mathbf{x})\left(2 - (\frac{x_1}{g(\mathbf{X})})^2\right)$	$g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i^2$	$[0.01, 1]\times[-1, 1]^{29}$
ZDT3	$f_1(\mathbf{x}) = x_1, f_2(\mathbf{x}) = g(x)\left(2 - \sqrt{\frac{x_1}{g(\mathbf{X})}} - \frac{x_1}{g(\mathbf{X})}\sin(10\pi x_1)\right)$	$g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i^2$	$[0.01, 1]\times[-1, 1]^{29}$
ZDT4	$f_1(\mathbf{x}) = x_1, f_2(\mathbf{x}) = g(\mathbf{x})\left(2 - \sqrt{\frac{x_1}{g(\mathbf{X})}}\right)$	$g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10\cos(4\pi x_i))$	$[0.01, 1]\times[-5, 5]^{9}$
ZDT6	$f_1(\mathbf{x}) = 1 - \exp(-4x_1)\sin^6(4\pi x_1), f_2(\mathbf{x}) = g(\mathbf{x})\left(2 - \left(\frac{f_1(\mathbf{X})}{g(\mathbf{X})}\right)^2\right)$	$g(x) = 1 + 9(\frac{1}{n-1}\sum_{i=2}^{n} x_i^2)^{0.25}$	$[0.01, 1]\times[-1, 1]^{9}$

The modified ZDT1 problem has a convex Pareto-optimal front for which solutions correspond to $0.01 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \ldots, 30$. Figure 1 shows the performance of all the algorithms after 5000 function evaluations. Table 2 shows the binary ϵ indicator values of all the algorithms on this problem. An element (i, j) in this table represents I_ϵ(algorithm j, algorithm i). Given two outcomes A and B, of different algorithms, the binary ϵ indicator $I_\epsilon(A, B)$ gives the factor by which an approximation set is worse than another with respect to all objectives. If $I_\epsilon(A, B) \leq 1$ and $I_\epsilon(B, A) > 1$ occurs then we an conclude that Algorithm A better than Algorithm B. These conditions are quite difficult to satisfy using binary ϵ indicator values. We will use the binary ϵ indicator values to conclude *partial* results: we will say that Algorithm A is *relatively better* than Algorithm B if $I_\epsilon(A, B) \leq 1.05$ and $I_\epsilon(B, A) > 1.05$. If $I_\epsilon(A, B) \leq 1$ and $I_\epsilon(B, A) > 1$ occurs the we say that Algorithm A is *definitively better* than Algorithm B. From Table 2 we obtain that with respect to binary ϵ metric all the three algorithms are incomparable. However, with respect to diversity S-NSGA-II performs the best (Table 4) while T-NSGA-II performs best in terms of convergence (Table 5).

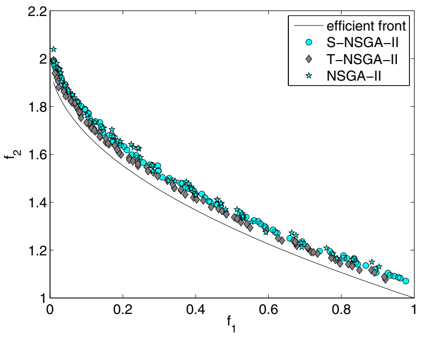

Fig. 1. Performance of the three algorithms on ZDT1

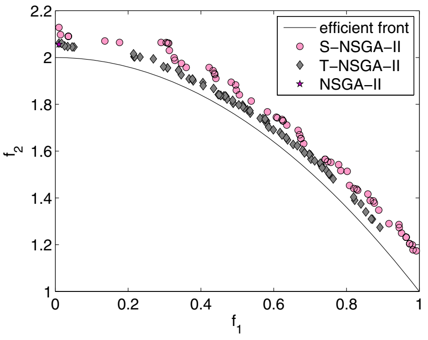

Fig. 2. Performance of the three algorithms on ZDT2

The modified ZDT2 problem has a non-convex Pareto-optimal front for which solutions correspond to $0.01 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \ldots, 30$. In the case of TPM algorithm (using a limited parametric study) we choose $C = 4.0$ Figure 2 shows the performance of all the algorithms after 5000 function evaluations. It can be seen both T-NSGA-II and S-NSGA-II find much more non-dominated solutions close to efficient front than NSGA-II. Although from Table 3 we observe that NSGA-II is *relatively better* than other two algorithms, however this happens since NSGA-II find *one* solution close to efficient front (while S-NSGA-II and T-NSGA-II find 38 and 37 solutions respectively). As can be seen from tables 4 and 5 that spread of both S-NSGA-II and T-NSGA-II is better than NSGA-II while convergence of T-NSGA-II is best. Next we consider ZDT3, this problem

Table 2. Binary ϵ indicator values on ZDT1

	SSM	TPM	NSGA
S-NSGA-II	1.0000	1.0124	1.0558
T-NSGA-II	1.0374	1.0600	1.0494
NSGA-II	1.0160	1.0056	1.0000

Table 3. Binary ϵ indicator values on ZDT2

	SSM	TPM	NSGA
S-NSGA-II	1.0000	1.0854	1.7517
T-NSGA-II	1.0512	1.0000	1.6139
NSGA-II	1.0353	1.0027	1.0000

has a convex discontinuous efficient frontier. Figure 3 shows the performance of all the algorithms after 5000 function evaluations. It can be seen that all the algorithms find non-dominated solutions close to efficient front in its all the 5 disconnected parts. From Table 6 one infers that S-NSGA-II performs relatively better than original NSGA-II, while T-NSGA-II and NSGA-II are incomparable. As can be seen from tables 4 and 5 diversity of T-NSGA-II is the best, while convergence of S-NSGA-II is best.

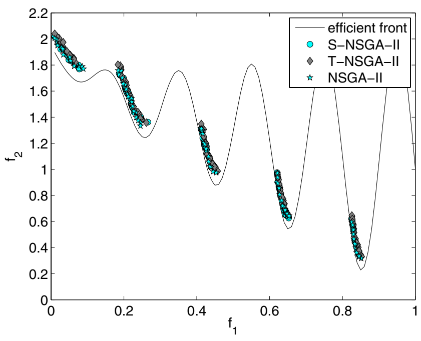

Fig. 3. Performance of the three algorithms on ZDT3

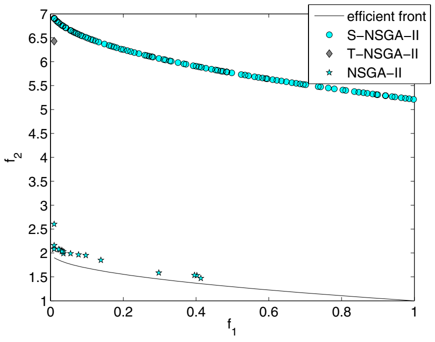

Fig. 4. Performance of the three algorithms on ZDT4

Table 4. Spread metric values. Best values are shown in bold.

	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
S-NSGA-II	**0.4992**	**0.7299**	0.8419	**0.8900**	0.5084
T-NSGA-II	0.6002	0.8018	**0.7914**	1.0000	**0.3799**
NSGA-II	0.7196	1.000	0.7998	1.0817	0.6267

The problem ZDT4 has a total of 100 distinct local efficient fronts in the objective space. The global Pareto-optimal solutions correspond to $0.01 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \ldots, 10$. Since ZDT4 is a complex multi-modal problem in this problem all the algorithms are run till 15000 function evaluations. Figure 4 shows the performance of all the algorithms. It can be seen that only NSGA-II is able to overcome many local Pareto optimal fronts and also performs best in terms of convergence (Table 5).

Next we consider another difficult problem, ZDT6. This problem has a non-convex and non-uniformly spaced Pareto-optimal solutions. The Pareto-optimal solutions correspond to $0.01 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \ldots, 10$. In case of NSGA-II we use $\eta_c = 1$ and $\eta_m = 1$ as these are more efficient for such difficult problems. Similarly, in the case of T-NSGA-II algorithm we choose $c = 0.5$ while in case of S-NSGA-II we use the same parameters. ZDT6 is a hard multi-objective problems and thus we run the simulations for a total of 15000 generations for each algorithm. Figure 5 shows the performance of all the algorithms after 15000

Table 5. Generational distance metric values. Best values are shown in bold.

	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
S-NSGA-II	0.0521	0.0825	**0.0267**	4.1118	0.0354
T-NSGA-II	**0.0342**	**0.0407**	0.0388	4.5287	**0.0214**
NSGA-II	0.0584	0.0562	0.0283	**0.1673**	0.1864

Table 6. Binary ϵ indicator values on ZDT3

	SSM	TPM	NSGA
S-NSGA-II	1.0000	1.0454	1.02810
T-NSGA-II	1.0160	1.0000	1.0113
NSGA-II	1.0620	1.0452	1.0000

Table 7. Binary ϵ indicator values on ZDT6

	SSM	TPM	NSGA
S-NSGA-II	1.0000	1.0028	1.2412
T-NSGA-II	1.0310	1.0000	1.2797
NSGA-II	1.0000	1.0000	1.0000

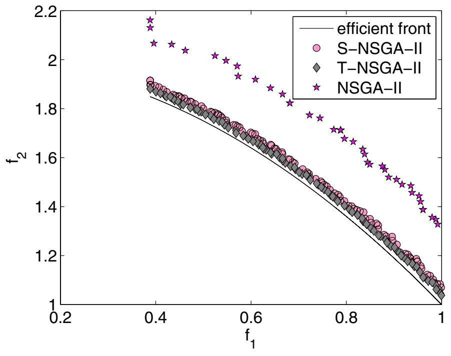

Fig. 5. Performance of the three algorithms on ZDT6

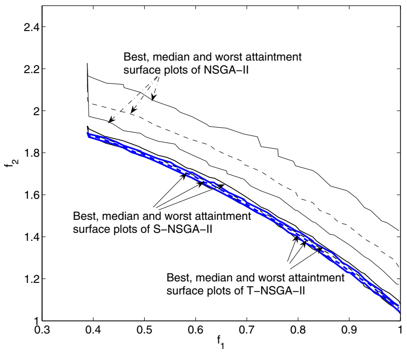

Fig. 6. Attainment surface plots of all the algorithms

function evaluations. It can be easily seen that both S-NSGA-II and T-NSGA-II perform better than NSGA-II. Also from Table 8 it can be inferred that performance of both S-NSGA-II and T-NSGA-II is *definitively better* than NSGA-II. The algorithms S-NSGA-II and T-NSGA-II are also better than NSGA-II in terms of diversity (Table 4) and convergence (Table 5). In order to address the issue of stochasticity on this hard problem we perform 21 different runs of all the algorithms and plot the best, median and worst attainment surface plot. Figure 6 shows the plots. One observes that even the worst attaintment surfaces of both S-NSGA-II and T-NSGA-II are much better than that of NSGA-II.

4 Conclusions

On a number of test problems of varying complexity, it has been observed that using the TPM and SSM as mutation search operators in NSGA-II performs better than the standard NSGA-II approach for a wide class of problems. However, for problems having multi-modal efficient front the above classical search techniques have not performed well. However this does not mean that they are not suited for solving multi-modal problems. Take for example, TPM in this the parameter C is crucial to get to the global efficient front. Once TPM gets stuck one strategy could be to restart the series by taking some larger value of C. SSM on the hand has a global search operator built in that and thus better numerical methods of solving the stochastic differential equation should be taken to see its performance. S-NSGA-II and T-NSGA-II perform substantially well over the evolutionary method in case of hard problems having a non-uniform density of solutions in the objective space. On the other hand, on all other problems considered here, the S-NSGA-II, has performed well in achieving both convergence and diversity of solutions with the *same* parameter values in all the test problems.

As an extension to this study one could use classical techniques for box-constrained [12], nonlinear inequality constrained [11,6], and non-differentiable [10] problems and try to combine with evolutionary optimization to create powerful hybrid algorithms.

Acknowledgements. The author acknowledges the partial financial support by the Gottlieb-Daimler- and Karl Benz-Foundation.

References

1. Peter A. N. Bosman and Edwin D. de Jong. Exploiting gradient information in numerical multi–objective evolutionary optimization. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 755–762, New York, NY, USA, 2005. ACM Press.
2. Peter A. N. Bosman and Edwin D. de Jong. Combining gradient techniques for numerical multi-objective evolutionary optimization. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 627–634, New York, NY, USA, 2006. ACM Press.
3. Martin Brown and Robert E. Smith. Effective use of directional information in multi-objective evolutionary computation. In *GECCO '03: Proceedings of the 5th annual conference on Genetic and evolutionary computation*, pages 778–789, 2003.
4. K. Deb. *Multi-objective optimization using evolutionary algorithms.* Chichester, UK: Wiley, 2001.
5. K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
6. Jörg Fliege and Benar Fux Svaiter. Steepest descent methods for multicriteria optimization. *Math. Methods Oper. Res.*, 51(3):479–494, 2000.
7. C. M. Fonesca and P. J. Fleming. On the performance assessment and comparison of stochastic multiobjective optimizers. In *Proceedings of Parallel Problem Solving from Nature IV (PPSN-IV)*, pages 584–593, 1996.
8. Ken Harada, Kokolo Ikeda, and Shigenobu Kobayashi. Hybridization of genetic algorithm and local search in multiobjective function optimization: recommendation

of ga then ls. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 667–674, New York, NY, USA, 2006. ACM Press.

9. Ken Harada, Jun Sakuma, and Shigenobu Kobayashi. Local search for multiobjective function optimization: pareto descent method. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 659–666, New York, NY, USA, 2006. ACM Press.
10. K. C. Kiwiel. Descent methods for nonsmooth convex constrained minimization. In *Nondifferentiable optimization: motivations and applications (Sopron, 1984)*, volume 255 of *Lecture Notes in Econom. and Math. Systems*, pages 203–214. Springer, Berlin, 1985.
11. H. Mukai. Algorithms for multicriterion optimization. *IEEE Trans. Automat. Control*, 25(2):177–186, 1980.
12. Maria Cristina Recchioni. A path following method for box-constrained multiobjective optimization with applications to goal programming problems. *Math. Methods Oper. Res.*, 58(1):69–85, 2003.
13. S. Schäffler, R. Schultz, and K. Weinzierl. Stochastic method for the solution of unconstrained vector optimization problems. *Journal of Optimization Theory and Applications*, 114(1):209–222, 2002.
14. P. K. Shukla, K. Deb, and S. Tiwari. Comparing Classical Generating Methods with an Evolutionary Multi-objective Optimization Method. In C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, pages 311–325, Guanajuato, México, March 2005. Springer. Lecture Notes in Computer Science Vol. 3410.
15. J. C. Spall. Implementation of the Simultaneous Perturbation Algorithm for Stochastic Optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 34(3):817–823, 1998.
16. G. Timmel. Ein stochastisches Suchverrahren zur Bestimmung der optimalen Kompromilsungen bei statischen polzkriteriellen Optimierungsaufgaben. *Wiss. Z. TH Ilmenau*, 26(5):159–174, 1980.
17. G. Timmel. Modifikation eines statistischen Suchverfahrens der Vektoroptimierung. *Wiss. Z. TH Ilmenau*, 28(6):139–148, 1982.
18. Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert da Fonseca. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, April 2003.