New Evolutionary Algorithm for Multi-objective Optimization and its Application to Engineering Design Problems

Dirk Büche [†] and Rolf Dornberger [‡]

[†]Alstom Power Technology, Segelhof 1, 5405 Baden-Dättwil, Switzerland, Email: Dirk.Bueche@power.alstom.com
[‡]Alstom Power Technology, now with Atraxis, Technology and Innovation, P.O. Box, 8058 Zurich Airport, Switzerland, Email: rdornber@atraxis.com

1 Abstract

Multi-objective optimization addresses problems with several design objectives, which are often conflicting, placing different demands on the design variables. In contradiction to traditional optimization methods, which combine all objectives into a single figure of merit, parallel optimization strategies such as evolutionary algorithms allow direct convergence to the Pareto front. This permits a designer to choose a posteriori a design from a variety of Pareto-optimal solutions.
In this paper, we introduce a new evolutionary approach called the subdivision method (SDM), using geometrical relations for the fitness assignment in the population, which guaranty diversity preservation and fast convergence to the Pareto set. New properties of the SDM are the constraint-free integration of target objective regions, and the ability to converge to the nondominated bounds of the objective space (Pareto solutions) as well as to the dominated bounds. The algorithm is applied to structural optimization problems and compared with state-of-the-art optimization algorithms.

2 Keywords

Multi-objective optimization, evolutionary algorithms, Pareto-optimization, subdivision method

3 Introduction

Today's design processes are increasingly complex and interdisciplinary tasks involving large design spaces, multiple objectives, and constraints. Finding a good solution is difficult, since objectives are usually conflicting and there exists no best solution to the problem, but a set of Pareto solutions representing the best compromise among the objectives. Pareto solutions are characterized that no other solution exists being superior in all objectives. Nevertheless, a decision-maker has come up with one design and has to choose the Pareto solution that fits best to his needs.
For solving multi-objective problems, several approaches are defined in literature. Horn [1] presents an overview and relates optimization strategies. Different approaches can be distinguished by the point of time, a decision maker introduces knowledge about the different objectives to the design process.
One approach is the multicriterion decision making before search. All objectives are aggregated into a single figure of merit. Examples for aggregation methods are weighted-sum, constraint approach or lexicographic ordering. Aggregation is mainly used for gradient-based strategies, which search in a point-to-point order.
A second approach is the multicriterion decision making after search. With no a priori knowledge about the objectives, an optimization run is executed converging towards the set of Pareto solutions. From the set, the decision-maker chooses the solution that is in his opinion the best compromise.
In this paper, we focus on the second approach. The first section of this paper performs a short overview on current state-of-the-art multi-objective evolutionary algorithms and presents a new method called the subdivision method. Preliminary versions of this algorithm have successfully been applied to multi-disciplinary problems in the field of turbomachinery design [2] [3]. Its features are now described in detail. The second part of this paper benchmarks the algorithm and analyzes the special features of the method.

4 Multi-objective Optimization Algorithms

Much effort has been spent over the past twenty years in the development and application of evolutionary algorithms for multi-objective optimization. Their fundamental property of population-based parallel search is one of the main reasons for their success in multi-objective optimization. It allows a parallel convergence towards the Pareto front within a single run.
Algorithms differ mainly in their selection operator, while recombination and mutation operators are usually taken from standard (single-objective) algorithms. Two main goals have to be considered for the selection operator: in average, superior solutions have to be selected and diversity must be preserved in the population. Both goals are needed in order to converge and to avoid concentration to small regions of the Pareto front, respectively. The latter is known as genetic drift [4]. An overview on state-of-the-art evolutionary algorithms can be found in Van Veldhuizen et.al. [5]. A detailed list of references for papers is presented by Coello on the web [6].
According to Van Veldhuizen [5], Pareto-based selection is most popular. Pareto-based selection uses the dominance criteria for the fitness assignment. In average, solutions that are dominated by fewer solutions are assigned a higher fitness and therefore have a higher probability to reproduce. Two promising algorithms are the Nondominated Sorting Genetic Algorithm (NSGA) [7] and the Strength Pareto Evolutionary Algorithm (SPEA) [8]. Both have been analyzed by many authors and are commonly used in applications [9].
NSGA assigns fitness by sorting the solutions in layers of nondominated solutions and uses niching to preserve diversity. SPEA uses the nondominated solutions for the fitness assignment and to preserve diversity. These two algorithms are later referred for comparison with the new subdivision method.

5 Subdivision Method

In the following, a new method for multi-objective optimization is presented called the Subdivision Method (SDM). The SDM bases on

an evolutionary algorithm with a special formulation for the fitness assignment and selection. Instead of using the dominance criterion like NSGA and SPEA, the SDM computes fitness locally in a subdivided objective space. It preserves diversity by selecting solutions uniformly out of all subdivisions.
In a first step, SDM is explained for the 2-objective case and then transferred to an arbitrary number of objectives. Certain features are further discussed.

5.1 Basic Algorithm for the 2 Objective Case

Consider the objective space of the two-objective minimization problem shown in Figure 1a. The objective space consists out of a feasible solution space, which is in this example completely bounded. The bounds are distinguished in nondominated and dominated areas. The main goal of the optimization process is to converge towards the nondominated area, representing the Pareto optimal and therefore best solutions to the problem. SDM proposes to divide the objective space into intervals along one objective axes. Figure 1b illustrates two intervals taken along objective 1. In the figure, an exemplary population of 6 solutions is given. A certain number of solutions can be found in each interval. The SDM performs a separate (local) selection in each interval. All individuals compete on behalf of objective 2. The best solution in the interval is selected, i.e. solution 5 and 6 are selected in the first and second interval, respectively. Then, the same process is repeated, but this time using intervals along objective 2 and considering objective 1 as selection criterion. The unification of the solutions of the both selections is the parent population for the next generation.
This selection treats every interval equally by selecting exactly one individual, independent on the number of individuals in an interval, and therefore avoids genetic drift.

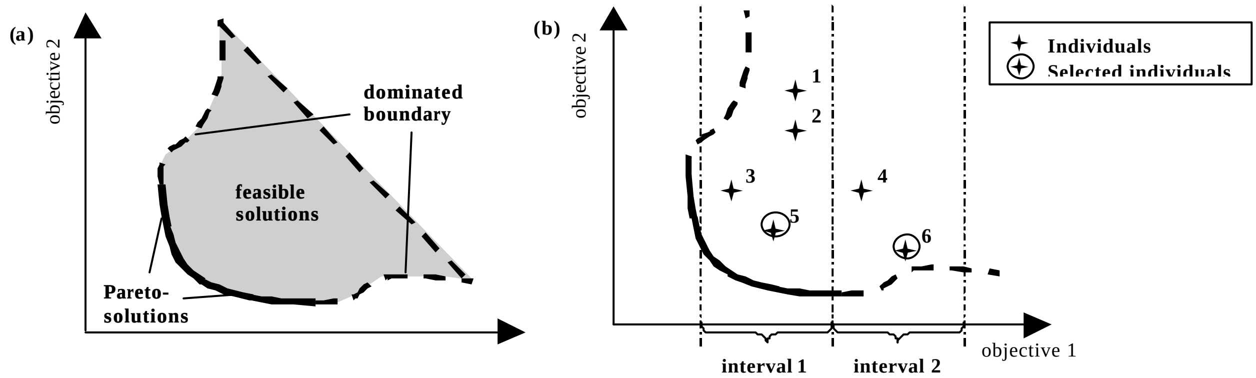


Figure 1: Objective space of a two-objective optimization problem (a) and illustration of the subdivision method (b).

5.2 Basic Algorithm for an Arbitrary Number of Objectives

Now the subdivision method is extended to an arbitrary number of objectives. Consider an objective space of M objectives. Each objective axis i is subdivided into k_i intervals. The SDM chooses one objective as selection criterion and the remaining M-1 objectives are subdivided into intervals. The algorithm can be explained in the following steps:

Algorithm 1:

1. Consider a current population P.
2. Generate an empty parent population P'.
3. Subdivide all objective axes in intervals.
4. Choose one of the M objectives as selection criterion and the remaining M-1 objectives as auxiliary objectives.
5. Generate an empty population P^{temp}.
6. Select one individual from P.
7. Find the interval in which the individual is located for all auxiliary objective axes.
8. Check if another individual exists in P^{temp}, which is located in the identical intervals. If no other individual exists, store the individual in P^{temp}. If one exists, store the superior with respect to the selection criterion and remove the inferior.
9. Repeat with step 6. until all individuals of the population P are considered once.
10. Add population P^{temp} to the parent population P'.
11. Repeat with step 4. until all objectives have been considered once as selection criterion.

The maximal size of the population P^{temp} can be computed by the product of the intervals for the auxiliary objective axis. An upper bound S for the size of the parent population P' can be computed by the sum of all populations P^{temp}:

$$S = \sum_{i=1..M} \left(\prod_{j=1...M,\, j \neq i} (k_j) \right) \tag{1}$$

Setting an equal number of intervals for all objectives, Equ. 1 simplifies to:

$$S = M \cdot k^{M-1} \tag{2}$$

5.3 Setting the Intervals

An essential issue in the SDM is right setting of the intervals. The intervals can be determined by specifying a lower and upper bound for the objective value and the number of intervals. The intervals may be equally spaced. For specifying the bounds, two different methods are proposed:

As a first method, the bounds can be defined by the user. This is useful if the target objective ranges are known. Often they are known from previous optimization runs. In addition, some objective ranges are known without previous calculations. For example, if the objective is the stress in a body, than the bounds may be set equal to the admissible stress.

A self-adaptive setting of the bounds is a second method. Among the population and their parents, all Pareto solutions are searched. The bounds for the intervals are set equal to the Pareto solution with the lowest and highest value for each objective.

While a user-input allows a constraint-free formulation of a particular target objective region, the self-adaptation is used for convergence towards the whole Pareto front.

The number of intervals is dependent on the desired number of parents. The relation between the number of parents and intervals is given in Equ. 1. The equation does not specify the exact number of parents but gives an upper bound, since no always all intervals are filled with solutions. Especially if the interval bounds are user-defined and represent target regions for the optimization process only a limited number of intervals will be filled in the beginning of the optimization run. For this case, an additional mechanism is implemented. If a minimal number of parents is not obtained from the selection, further intervals are added by increasing the interval bounds and the selection process is restarted. This is repeated until a minimal number of parents is reached.

The interval spacing is also a free parameter for the user. The interval spacing may be equidistant or logarithmically distributed, in order to select uniformly or to refine the intervals for small objective values, respectively.

5.4 Elitism

Elitism is a technique of conserving the best solutions obtained so far in the optimization process. For multi-objective optimization, elitism is seen as a necessity [1][5]. For Pareto-based algorithms like NSGA and SPEA, elitism is usually performed by storing the nondominated solutions in a secondary population [5]. The secondary population is updated with the current population by adding new nondominated solutions and removing dominated.

The subdivision method does not store elite solutions. Instead, the selection process considers all solutions computed so far. This is possible since the computational cost of the selection is low and linear to the number of individuals, as will be shown in the next section. Assigning a limited lifetime to the solutions is reducing the computation costs and limiting the impact of a solution.

5.5 Computation Costs

For most applications, the computational cost of the optimization algorithm can be neglected compared to the cost of the application. Nevertheless, costs are analyzed in literature. Van Veldhuizen [5] computed the costs for different of Pareto-based multi-objective algorithms. According to him, the costs of Pareto-based selection schemes like SPEA and NSGA is proportional to $(N+N_1)^2$, where N and N_1 are the size of the population and secondary population, respectively.

For the SDM, the computational are computed for a population of size N as follows: According to the algorithm in Section 5.2, each of the M objectives is chosen once as selection criterion. A solution has to be sorted into the k_i intervals of the auxiliary (M-1) objectives. Then the solution is compared to best solution already found for this combination of intervals. In total, an upper bound for the computational cost C is equal to the product of these steps, given by:

$$C = N \cdot \sum_{i=1\ldots M} \left(\sum_{j=1\ldots M,\, j\neq i} (k_j) + 1 \right) = N \cdot (M-1) \cdot \left(\sum_{j=1\ldots M} (k_j) + 1 \right) \tag{3}$$

The cost is linear in the population size N, quadratic with the number of objectives and linear in the number of intervals along the objective axis. For equal number of intervals for all objectives, Equ. 3 simplifies to:

$$C = N \cdot (M-1) \cdot M \cdot (k+1) \tag{4}$$

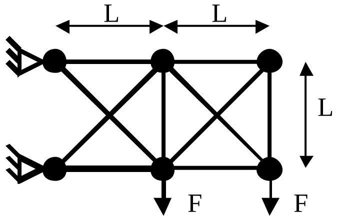
The linearity of the cost over the population size is an advantage compared to the Pareto-based algorithms, for which cost increases quadratic over the population size.

6 Applications

In the following two structural optimization problems are presented. The first problem is the optimization of a 2D truss with two objectives and one constraint. It is used to compare SDM with NSGA and SPEA. The second problem is a simple 1D tower with 3 objectives. This problem shows SDM for higher objective dimensions and illustrates its special properties.

6.1 Ten-bar Truss

A well-known structural design problem is the ten-bar truss [10] as shown in Figure 2. Two loads F are applied. The free design variables are the 10 cross-sections of the bars. The objectives are to minimize the total weight of the truss as well as to minimize largest deflection of the 4 unconstrained bar intersections. Constraints are applied to the admissible stress in the bars for tension and compression. The properties of the truss are taken from [10], but converted to the SI system of units and rounded. They are given in Figure 2. The stress in the bars and the nodal deflections are calculated by a standard Finite Element code for two-node bars and small deflections. The performance of the NSGA, SPEA and SDM is analyzed.



Truss properties		
Element length *L*	*9*	m
Young's modulus	70.000	N/mm^2
specific weight	2.700	kg/m^3
Loading F	45.000	N
Stress contraint	±170	N/mm^2
Cross-section	[50 26.000]	mm^2

Figure 2: Ten-bar truss problem

6.1.1 Optimizer Settings

For NSGA, SPEA and SDM, the population size is set to 60 and the optimization is limited to 2.000 computed solutions. Binary intermediate and uniform recombination is used with equal probability and the overall recombination probability is set to 60%. A normal distributed mutation is applied with a relative standard deviation of 10% of the design variable range and a mutation probability of 15% per design variable. For NSGA and SPEA a secondary population is used as described by Zitzler und Thiele [5] with the maximal size of 15 individuals. For the subdivision method, both objective axes are subdivided into 15 intervals and the lifetime of the individuals is not limited. SDM is executed with self-adapting subdivision bounds in order to allow a fair comparison with NSGA and SPEA and not introducing any a priori user knowledge.

6.1.2 Optimization Results

Figure 3a shows the Pareto solutions for NSGA, SPEA and SDM. The performance of the algorithms differs mainly for low weight structures, since from the problem definition this is more difficult to design. For this problem, SDM performs best and is dominating the Pareto front of NSGA and SPEA for low weights. SDM shows also a wider Pareto front.

6.1.3 SDM with User-defined Subdivisions

In Figure 3b the performance of SDM is shown for user-defined settings. The intervals were set between 1800 to 2500 kg for the weight objective and between 40 to 60 mm for the displacement objective. The figure shows all evaluated solutions. A clear concentration of solutions occurs in the regions of the user-defined intervals. Comparing the result with the self-adaptive setting in Figure 3a, the convergence improved within the user defined bounds.

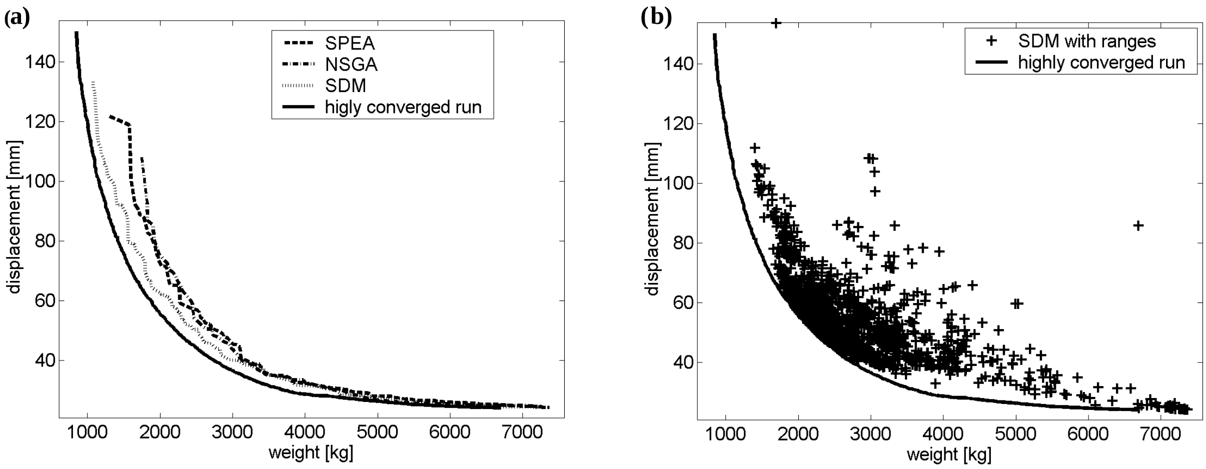


Figure 3: Ten-bar truss problem: Pareto-front after 2.000 computations for NSGA, NPEA and SDM (a) and convergence of SDM for user-defined objective bounds (b).

6.2 Brick Tower

A brick tower is considered as a second structural optimization problem and is illustrated in Figure 4. The tower consists of six cylindrical bricks. All bricks are of the same height *h* and density ?. The tower is loaded by its dead load.

The design variables of this problem are the cross-sections *A* of the bricks. All necessary data is given in Figure 4. The objectives are to minimize weight, stress and fundamental frequency of the tower.

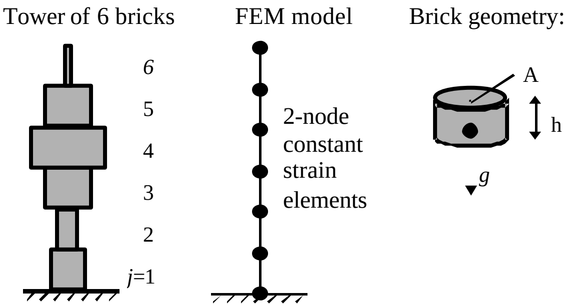
The weight *W* of the tower is calculated as the sum of the brick weight by:

$$W = \sum_{i=1}^{6} rhgA_i \quad (5)$$

Stresses are computed at bottom of each brick by summing up the weights of the bricks on top. Stress concentration factors are neglected. The largest stress in the bricks is considered as objective and is given by:

$$s_{max} = \max_{j} \left(\frac{r h}{A_j} \sum_{i=j}^{6} A_i \right) \qquad (6)$$

The vibration of the tower is approximated by a 1D model in vertical direction. Each brick is discretized by a 2 node constant strain finite element. The fundamental eigenfrequency is computed as the lowest frequency of the 6 possible eigenmodes of the tower.



Truss properties		
Brick height h	100/6	m
Young's modulus E	$2\ 10^5$	N/mm^2
Gravity constant g	10	m/s^2
specific mass ?	10^4	kg/m^3
Cross-section A	[1 10]	m^2

Figure 4: Brick tower problem: Tower consisting of 6 bricks with variable cross-section and is optimized for minimal stress, weight and fundamental frequency

6.2.1 Optimizer Settings

The aim of the brick tower is give insights in the features of the SDM and not be used to benchmark different algorithms. Therefore, NSGA and SPEA are not considered here. The application underlines the role of user defined objective bounds.

For this optimization problem, we simplify Algorithm 1 by just considering one objective as selection criterion instead of all 3 objectives. The fundamental frequency is chosen as selection criterion, while the maximal stress in the bricks and the truss weight are chosen as auxiliary objectives. Stress and weight are chosen as auxiliary objectives, since bounds can easily be defined for them.

The exact lower and upper bound for the weight of the tower is obtained by setting all bricks to the smallest and largest cross-section, respectively. This results in truss weights of 10^7 and 10^8 *N*. A save lower bound assumption for the stress in the bricks is given by computing the stress by a single brick instead of the whole tower. This is area independent and leads to a stress of 1,67 *N*. A save upper bound for the stress is given by dividing the maximal truss weight through the minimal cross-section, leading to a highest stress of 100 *N*.

For both auxiliary objectives, 25 intervals were chosen. For the weight axis the intervals are set equidistant between the bounds and for the stress axis, the intervals are exponentially growing over the axis. The intervals are shown in Figure 5. The optimization process is limited to 20.000 evaluations and a population of 300 individuals is considered.

6.2.2 Optimization Result

The results of this optimization problem are shown in Figure 5. The weight and stress objectives are plotted in the paper space, while the fundamental frequency is given by the color. The figure contains a surface that is obtained by connecting always the individual with the lowest fundamental frequency of each rectangle. The figure illustrates the conflict in minimizing all 3 objectives. Four characteristic designs are given in the right part of the figure. They illustrate the extreme solutions to the problem. It can be seen that for a high stress and low frequency, the thinnest bricks are at the bottom and the thicker ones are at the top of the tower. For a low stress and high frequency it is vice versa.

Each solution in a rectangle is optimal in the sense that no other solution was found with lower minimal fundamental frequency for this rectangle. However, this does not include Pareto optimality. Selecting the Pareto-ideal solutions among all computed leads to the surface shown in Figure 6. The Pareto front is incoherent and represents a subset of the solutions in Figure 5. The Pareto-front would be the result of Pareto-based optimization strategies like NSGA or SPEA. The result of the subdivision method in Figure 5 was able to find the Pareto-ideal solutions. In addition, it provides for each possible combination of truss weight and stress the solution with minimal fundamental frequency. These solutions represent the dominated and nondominated bound of the of the feasible objective space for minimizing the fundamental frequency.

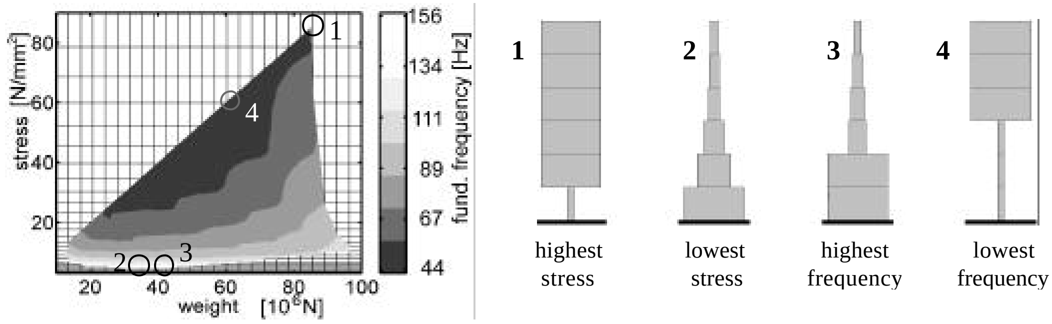


Figure 5: Three-objective brick tower problem: Solutions of the subdivision method for the weight, stress and fundamental frequency optimization of a brick tower: The 3-dimensional surface was created by connecting always the best solution of each rectangle. The subdivisions are added as thin lines. The four brick towers on the bottom of the picture illustrate the shape of the tower for the extreme solutions.

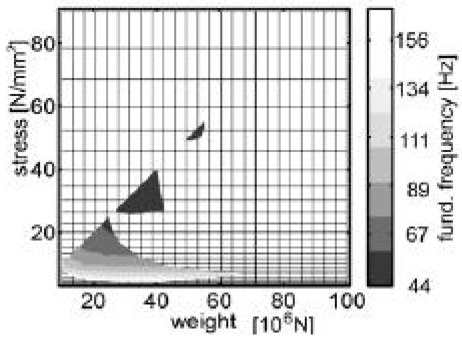


Figure 6: Three-objective brick tower problem: Pareto-front for the weight, stress and fundamental frequency optimization of a brick tower

7 Conclusions

This work introduced a new multi-objective selection algorithm, which uses a geometrical subdivision of the objective space and performs local selection. It allows a constraint free integration to user-defined target objective regions and a self-adaptive setting for converging towards the Pareto ideal solutions. The user-defined approach is important if certain minimal requirement to the objective values exist. It converges towards the dominated and nondominated bound of the feasible solution space in the target region.
The self-adaptive setting was compared with two state-of-the-art algorithms for a two-objective truss problem and showed excellent convergence properties towards the whole Pareto front.
The SDM is simple to implement. A short algorithm is given in this paper and modern aspects for multi-objective optimization like elitism are described.
The SDM performs to a uniformly selection of solutions in the objective space, regardless if they are Pareto-ideal ore not. This is of particular interest for a disrupted, discontinuous Pareto front as obtained for the brick tower optimization. In addition, it allows the convergence to the dominated boundary of the objective space. It provides additional insights in the objective space and retrieves the definite bounds of the Pareto front, which is not always captured by Pareto-based methods.

8 Acknowledgement

The authors would like to thank the Institute of Computational Sciences at ETH Zurich for their support.

9 References

[1] Horn J. Multicriterion decision making, in T. Bäck, D.B. Fogel and Z. Michalewicz, Handbook of Evolutionary Computation, page F1.9: 1-15, Oxford University Press, New York, 1997

[2] Dornberger R, Büche D and Stoll P. Multidisciplinary optimization in turbomachinery design. Eccomas 2000, Barcelona, Spain, CMNM 102

[3] Dornberger R, Stoll P, Büche D and Neu A. Multidisciplinary turbomachinery blade design optimization. 38th AIAA Aerospace Sciences Meeting and Exhibit, 2000, Reno, USA, AIAA 2000-0838

[4] Goldberg D E, Segrest P. Finite markov chain analysis of genetic algorithms, Proceedings of the Second International Conference on Genetic Algorithms, Lawrence Erlbaum, 1987, p. 1-8

[5] Van Veldhuizen, D A and Lamont G B. Multiobjective evolutionary algorithms: analyzing state-of-the-art. Evolutionary Computing, 2000, MIT Press, Cambridge, MA, 8(2): 125-147

[6] List of references and some papers online at: http://www.lania.mx/~ccoello/EMOO/EMOObib.html

[7] Srinivas N and Deb K. Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation, 1994, (2) 3:221-248

[8] Zitzler E and Thiele L. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Transactions on Evolutionary Computation, 1999, 3(4):257-271

[9] Zitzler E, Deb K, Thiele L, Coello Coello C A and Corne D (Eds.). Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization, EMO 2001, Zurich, Switzerland

[10] Crossley, Cook, Fanjoy and Venkayya, Using the two-branch tournament algorithm for multiobjective design. A Collection of technical Papers, 1998, 39th AIAA/ASME/ASCE/ AHS/ASC, 1998, Long Beach, California, Part 2, p.1752-1762