Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational and Applied Mathematics 168 (2004) 255–265
JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS
www.elsevier.com/locate/cam

Application of genetic algorithms to lubrication pump stacking design

V. Kelner*, O. Léonard

Turbomachinery Group, University of Liège, Chemin des Chevreuils 1, 4000 Liège, Belgium

Received 26 September 2002; received in revised form 27 May 2003

Abstract

Sizing a pump stacking used in an aircraft lubrication system is a challenging task. The combination of several pumps, in parallel and in a single casing, must deliver specified oil flow rates, on a variable number of circuits, and under given flight conditions. Furthermore, the optimal assembly has to minimize overall dimensions, weight and cost. This optimization problem involves a large space search, continuous and discrete variables and multi-objectives. Genetic Algorithms (GA)—stochastic search methods that mimic the metaphor of natural biological evolution—seem well suited to solve that kind of problems. A new GA is proposed. The efficiency of this GA is first demonstrated in solving various mathematical test-cases and then applied to the industrial problem.
© 2003 Elsevier B.V. All rights reserved.

Keywords: Multiobjective optimization; Constrained optimization; Mixed variables; Genetic algorithms; Optimal design; Aircraft lubrication

1. Mathematical statement of the optimal sizing problem

The aircraft lubrication system provides lubrication and cooling for all gears and bearings of the engine [15]. Generally, positive displacement pumps, such as Gerotor (Fig. 1) or gear pumps, are used both as pressure pumps (distribution of oil to all the lubricated parts) and scanvenge (return oil) pumps [4]. These pumps are incorporated in a common casing and must deliver specified oil flow rates under various flight conditions.

* Corresponding author.
E-mail address: v.kelner@ulg.ac.be (V. Kelner).

0377-0427/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2003.05.022

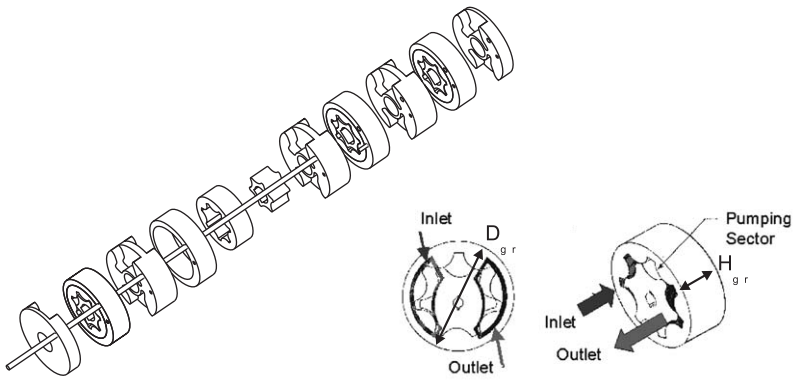

Fig. 1. Gerotor pump stacking.

In mathematical terms, the optimal sizing problem may be defined as finding $\vec{x}^*$ that minimizes

$$\vec{f}(\vec{x}) = [f_1(\vec{x}), \ldots, f_{n_{\text{of}}}(\vec{x})]^{\text{T}} \tag{1}$$

with

$$\vec{x} = [x_1, \ldots, x_{n_x}]^{\text{T}} \tag{2}$$

subject to

$$\vec{h}(\vec{x}) = [h_1(\vec{x}), \ldots, h_{n_h}(\vec{x})]^{\text{T}} = \vec{0}, \tag{3}$$

$$\vec{g}(\vec{x}) = [g_1(\vec{x}), \ldots, g_{n_g}(\vec{x})]^{\text{T}} \leqslant \vec{0}. \tag{4}$$

In this study, n_{p} pumps, driven by a unique shaft (Fig. 1), and n_{fc} flight conditions are considered. The design variables can be expressed as

$$\begin{aligned}\vec{x} &= [\vec{H}_{\text{gr}}, \vec{D}_{\text{gr}}, \vec{N}_{\text{t}}, \vec{M}_{\text{gr}}]^{\text{T}} \\ &= [H_{\text{gr}}^1, \ldots, H_{\text{gr}}^{n_{\text{p}}}, D_{\text{gr}}^1, \ldots, D_{\text{gr}}^{n_{\text{p}}}, N_{\text{t}}^1, \ldots, N_{\text{t}}^{n_{\text{p}}}, M_{\text{gr}}^1, \ldots, M_{\text{gr}}^{n_{\text{p}}}]^{\text{T}},\end{aligned} \tag{5}$$

where H_{gr}^j, D_{gr}^j, N_{t}^j and M_{gr}^j, respectively, denote the thickness, the diameter, the number of teeth in the inner rotor and the multiplicity of the jth Gerotor pump in the stacking.

As the pumps have to be chosen in a catalog, $\vec{D}_{\text{gr}}$ and $\vec{N}_{\text{t}}$ have discrete components in addition to $\vec{M}_{\text{gr}}$. On the other hand, $\vec{H}_{\text{gr}}$ is a set of continuous parameters.

According to the end-user, the three following constraints have to be satisfied:

- the pressure pump (identified by the upper script $j = 1$) must have a six teeth inner rotor

$$N_{\text{t}}^1 = 6, \tag{6}$$

- the thickness of each Gerotor must be in a defined range

$$\vec{H}_{\text{gr}}^{\min} \leqslant \vec{H}_{\text{gr}} \leqslant \vec{H}_{\text{gr}}^{\max}, \tag{7}$$

- each pump, for each flight condition, must deliver a specified minimum oil flow rate

$$q_{\min}^{i,j} \leqslant M_{\mathrm{gr}}^{j} q_{\mathrm{gr}}^{i,j}, \quad i = 1, \ldots, n_{\mathrm{fc}} \quad j = 1, \ldots, n_{\mathrm{p}}. \tag{8}$$

The optimal stacking has to minimize overall dimensions, weight and cost. This can be achieved by simultaneously minimizing the three following objective functions:

- the number of stacked Gerotor

$$N_{\mathrm{s}} = \sum_{j=1}^{n_{\mathrm{p}}} M_{\mathrm{gr}}^{j}, \tag{9}$$

- the height of the stacking

$$H_{\mathrm{s}} = \sum_{j=1}^{n_{\mathrm{p}}} M_{\mathrm{gr}}^{j} H_{\mathrm{gr}}^{j}, \tag{10}$$

- the volume of the stacking

$$V_{\mathrm{s}} = \sum_{j=1}^{n_{\mathrm{p}}} M_{\mathrm{gr}}^{j} H_{\mathrm{gr}}^{j} \frac{\pi (D_{\mathrm{gr}}^{j})^2}{4}. \tag{11}$$

This optimization problem clearly involves a large space search, mixed (continuous and discrete) variables and multi-objectives. Traditional first-order optimization methods based on the knowledge of the Jacobian could not be used. The industrial partner attempted to solve the problem with an expert system, but with a limited success and at the cost of a prohibitive computational effort.

2. Genetic algorithms

Genetic algorithms (GAs), first developed by Holland [9], are zero order search methods that mimic the Darwins's principles of natural selection and survival of the fittest. GAs work with artificial populations of individuals that represent candidate solutions and, in spite of their diversity, all GAs are based on the same iterative procedure [7] (Fig. 2).

Each item in Fig. 2 will be briefly presented hereafter, as well as their implementation in our GAGERO (Genetic Algorithm for GERotor Optimization).

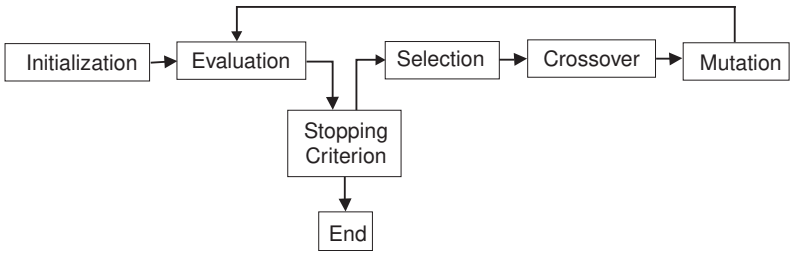

Fig. 2. Standard GA flowchart.

2.1. Basic mechanics of GAs

Initialization creates an initial population of N_{pop} by random. These candidate solutions are a set of chromosomes or strings of characters (letters and/or numbers) that represent the first potential solutions of the problem. To overcome the major drawbacks of a classical binary coding (huge string length, discrepancy between the binary and the real design space), a real-valued representation has been chosen.

During *evaluation*, the fitness of each candidate solution is computed by objective values.

Selection [1,8] is the process in which the fittest individuals are selected and reproduced (once or more) to create a new mating pool of N_{pop} parents. Two traditional selection schemes have been implemented in our code:

- Ranking selection [2]: the population is sorted from best (rank $R = 1$) to worst ($R = N_{\text{pop}}$). Then a linear selection probability p_{s} is computed for each individual by

$$p_{\text{s}}(i) = \frac{1}{N_{\text{pop}}}\left[S_p - 2(S_p - 1)\,\frac{R(i)}{N_{\text{pop}}}\right], \tag{12}$$

  where S_p (selection pressure) is a user specified parameter ($S_p \in [1, 2]$). Afterwards this probability distribution is sampled by a biased roulette wheel.
- Tournament selection: N_{tour} individuals are randomly selected in the population. The best one is reproduced in the new mating pool. The process is repeated N_{pop} times with reintegration of the winner in the population.

Crossover [13] provides the search mechanism of the GA. It generates children (new solutions) by exchanging features of the previously selected parents. Two traditional schemes have been implemented in our code:

- Blended crossover (BLX-α) [5] is applied to the continuous variables: parents $\vec{P}_1$, $\vec{P}_2$ give birth to children $\vec{C}_1$, $\vec{C}_2$ according to

$$\vec{C}_1 = \gamma\vec{P}_1 + (1 - \gamma)\vec{P}_2 \tag{13}$$

$$\vec{C}_2 = (1 - \gamma)\vec{P}_1 + \gamma\vec{P}_2 \tag{14}$$

  with

$$\gamma = (1 + 2\alpha)u - \alpha, \tag{15}$$

  where u is a random number in [1,2] and α is user-specified parameter.
- One-point crosser is used for the discrete part of the chromosome: the vector components of two parents are simply swapped in groups at a random position.

Mutation [13] randomly alters some individual's genes. In our application, mutation is only applied to the discrete variables.

2.2. GAs and multi-objective constrained optimization problems

As stated by Eqs. (1)–(4), a multi-objective optimization problem (MOP) can be viewed as the problem of finding—from among the set F of all solutions which satisfy (3) and (4)—the particular set $\vec{x}^*$ which yields the optimum values of all the objective functions [3].

However, in a MOP, there is rarely a single point that simultaneously optimizes *all* the objective functions, and therefore, the notion of "optimum" is quite different.

When dealing with MOP, the commonly adopted notion of optimality is the one proposed by Pareto [12]: a vector of decision variables $\vec{x}^*$ is *Pareto optimal* if there does not exist another $\vec{x} \in F$ such that $f_i(\vec{x}) \leqslant f_i(\vec{x}^*)$ for all $i = 1, \ldots, n_{\text{of}}$ and $f_j(\vec{x}) < f_j(\vec{x}^*)$ for at least one j.

According to this definition, a MOP has no unique solution, but rather a set of compromised solutions (called the Pareto front): these so called nondominated solutions are the ones for which the value of any objective functions cannot be improved without deteriorating at least one of the others.

A usual way to solve a MOP is to reduce it into a classical single objective problem through an aggregating method (weighted sum, goal programming, weighted min–max, etc.) [3].

For example, in a weighted sum approach, the objective function is given by

$$\Phi(\vec{x}) = \sum_{k=1}^{n_{\text{fo}}} w_k f_k(\vec{x}), \tag{16}$$

where $w_k \geqslant 0$ are the weighting coefficients representing the relative importance of the objectives $f_k(\vec{x})$.

The major drawback of these aggregating techniques is that the obtained solution corresponds to only one point of the Pareto front, depending on the arbitrary choice of the weights. Moreover, there is no guarantee that the non dominated solutions are uniformly distributed on the Pareto front.

On the contrary, GAs, working with a population of candidate solutions, are able to approximate the Pareto front in a single run. Furthermore, they are less susceptible to the shape or continuity of the Pareto front (they can approximate concave or noncontinuous Pareto front).

These advantages have made them very popular to solve *unconstrained* MOPs and numerous Pareto based approaches (MOGA, NSGA, NPGA, etc.) have been proposed and compared in the literature [3,16].

When the optimization problem involves constraints, the classical approach is to use a penalization method. In this technique, a new objective function is simply defined by adding a (static, dynamic or adaptive) penalty to each unfeasible solutions.

For example, in the dynamic penalty approach [10], the new objective is given by

$$\Phi'(\vec{x}) = \Phi(\vec{x}) + (C \cdot t)^{\alpha} \sum_{g_j > 0} [g_j(\vec{x})]^{\beta}, \tag{17}$$

where t is the generation number; C, α and β are the parameters of the method; and Φ is the objective function of the unconstrained problem.

The penalization techniques are very popular because they can be implemented without significant modification of the standard genetic algorithm. However, to be efficient, they require an adequate

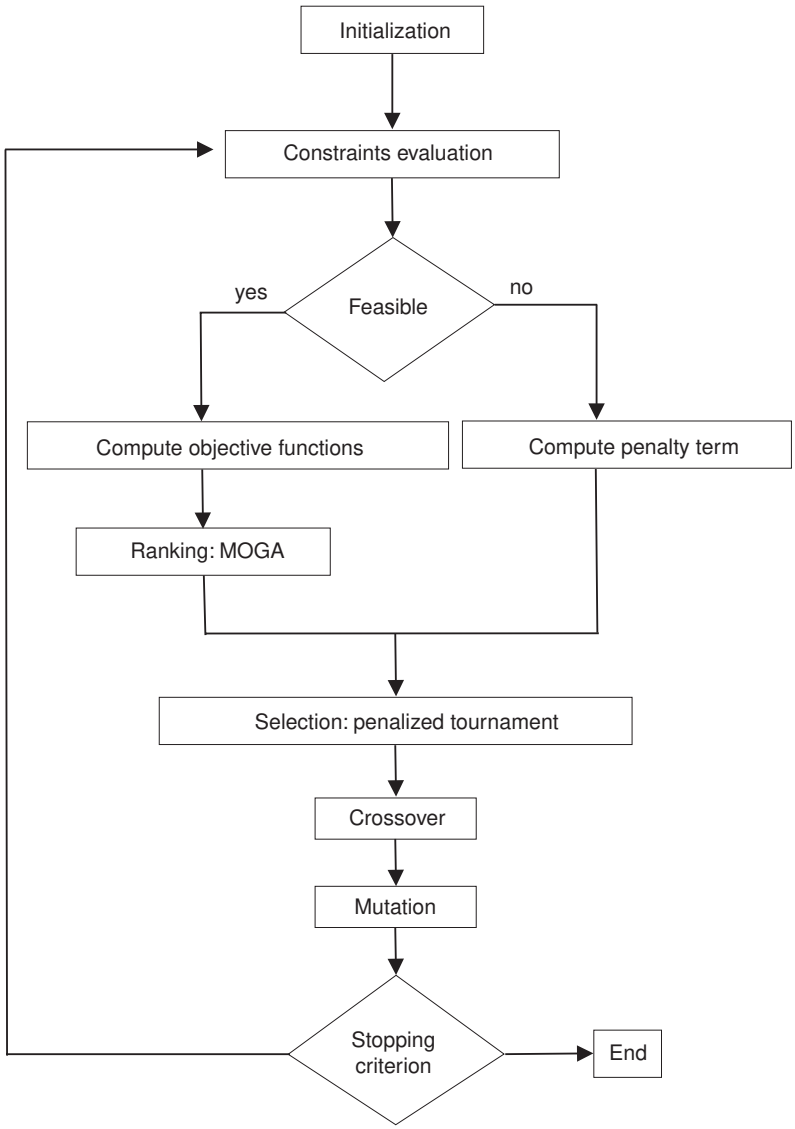

Fig. 3. GAGERO flowchart.

"tuning" of the parameters. Another major drawback of these methods is that they lead to consider once again a single objective approach: in the case of a MOP, the objective function Φ in Eq. (17) has to be computed through an aggregating method.

We propose here an original approach to take into account *simultaneously* the multiobjective and constrained aspects of the optimization problem.

In our application (Fig. 3), the constraints are firstly evaluated for each individual. On the one hand, the feasible solutions are ranked according to the MOGA algorithm [6]. This scheme (Fig. 4)

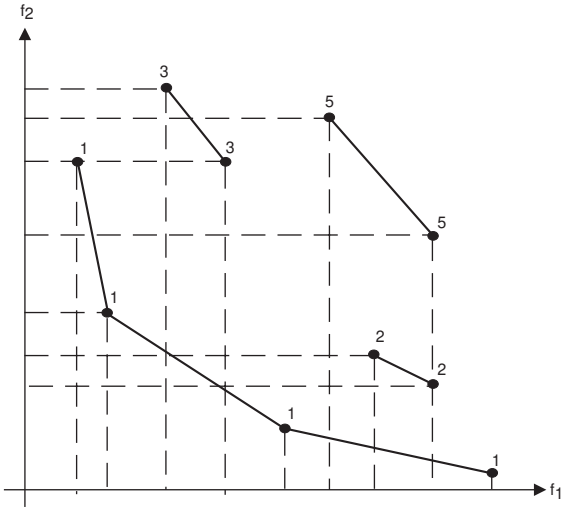

Fig. 4. Example of the MOGA ranking procedure with two objectives functions which have to be minimized.

consists in evaluating the individual's rank by

$$R(i) = 1 + p(i), \tag{18}$$

where $p(i)$ denotes the number of (feasible) solutions in the current population by which the individual is dominated.

On the other hand, each infeasible solution receives a penalty fitness computed by

$$R_{\text{const}} = \sum_{g_j > 0} g_j(\vec{x}). \tag{19}$$

At last, a selection, based on a "penalized tournament", is applied. It consists of randomly choosing and comparing the (generally two) individuals:

- if they are all feasible, the best ranked element wins,
- if they are all infeasible, the one having the lower R_{const} value wins,
- if one is feasible and the others are infeasible, the feasible individual wins.

The major advantage of this method is that the computation of the objective functions and the ranking procedure are performed only for the feasible solutions. Furthermore, this Pareto based technique can approximate the Pareto front in a single run. Moreover, with this approach, the user does not need to choose any explicit penalty parameter.

3. Mathematical test-cases

GAGERO has been tested on many mathematical problems. Two relevant test-cases proposed by Poloni are presented here.

The first test-case (TC1) [14] is a maximization of a two objective functions defined by

$$f_1(x_1,x_2) = -[1 + (A_1 - B_1)^2 + (A_2 - B_2)^2] \tag{20}$$

$$f_2(x_1,x_2) = -[1 + (x_1 + 3)^2 + (x_2 + 1)^2] \tag{21}$$

with

$$x_1,x_2 \in [-\pi, +\pi], \tag{22}$$

$$A_i = \sum_{j=1}^{2} a_{ij}\sin(\alpha_j) + b_{ij}\cos(\alpha_j), \tag{23}$$

$$B_i = \sum_{j=1}^{2} a_{ij}\sin(\beta_j) + b_{ij}\cos(\beta_j), \tag{24}$$

$$a = \begin{bmatrix} 0.5 & 1.0 \\ 1.5 & 2.0 \end{bmatrix}, \tag{25}$$

$$b = \begin{bmatrix} -2.0 & -1.5 \\ -1.0 & -0.5 \end{bmatrix}, \tag{26}$$

$$\alpha = [1.0\ \ 2.0], \tag{27}$$

$$\beta = [x_1\ \ x_2]. \tag{28}$$

The second test-case (TC2) proposed by Poloni [14] is an extension of the previous one where two constraints given by

$$(x_1 - 2)^2 + (x_2 - 2)^2 \leqslant 9, \tag{29}$$

$$(x_1 + 2)^2 + (x_2 + 2)^2 \geqslant 9 \tag{30}$$

have also to be satisfied.

3.1. Results

The parameters used in the algorithm are summarized hereafter:

- N_{pop}: population size—set to 576.
- S_p: selection pressure—set to 2.
- ε: the GA is stopped when the difference between the objective function of the best and the worst individual in the population is lower or equal to this parameter—not used for Poloni's test cases.
- $G_{\max}$: maximum number of allowed generations—set to 250.

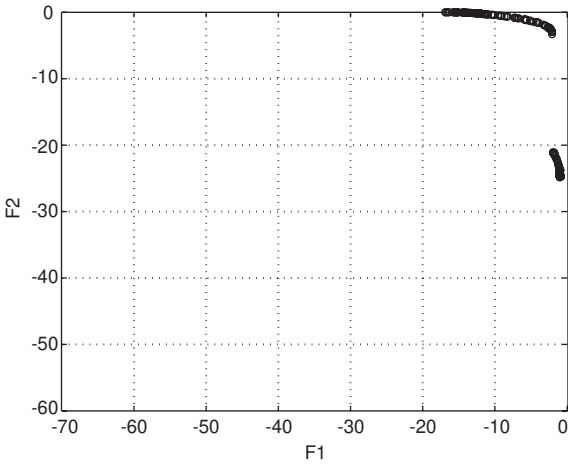

Fig. 5. Pareto front for TC1.

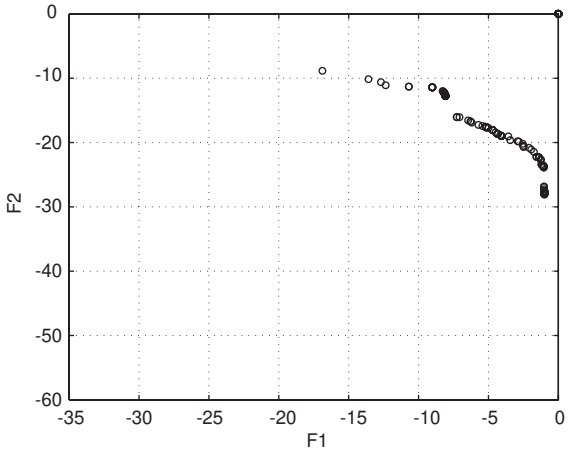

Fig. 6. Pareto front for TC2.

Figs. 5 and 6 show the Pareto fronts for the test-cases TC1 and TC2, respectively. GAGERO has clearly identified the set of non dominated solutions.

Moreover, the concave and discrete parts of the Pareto fronts have been uniformly approximated.

4. An industrial application

GAGERO has been applied to a real problem proposed by the industrial partner i.e., optimizing a lubrication pump stacking as described in Section 1.

The values for the discrete variables, such as the three different flight operating conditions and the five considered Gerotor pumps can be found in [11].

Table 1
Examples of optimal stacking

Pump (n°)	Optimal stacking ♮1				Optimal stacking ♮2			
	H_{gr} (mm)	D_{gr} (mm)	N_t	M_{gr}	H_{gr} (mm)	D_{gr} (mm)	N_t	M_{gr}
1	6.159	80.0	6	2	5.951	80.0	6	2
2	10.454	80.0	4	1	5.670	80.0	4	2
3	7.252	80.0	4	1	7.019	80.0	4	1
4	8.770	80.0	6	1	7.014	80.0	4	1
5	5.634	80.0	6	1	7.640	63.0	6	1

Various combinations of Gerotor pumps have been proposed by GAGERO.

Table 1 provide two examples of optimal stacking among the seven found by the code.

The results have been obtained after 20 iterations and with a initial population of 500 individuals.

This computation has been performed on a 600 MHz—256 Mb RAM—Pentium III and the CPU time was about 160 s.

5. Conclusions

A genetic algorithm (GA) has been developed based on a real coding that can easily deal with mixed (discrete and continuous) variables. This GA is able to solve effectively multiobjective *and* constrained optimization problems, through an original approach which combines MOGA and a penalized tournament selection scheme. This approach is very effective when most of the initial candidate solutions violate the constraints, as it is usually the case after a random exploration of the design space.

Results provided by GAGERO were obtained within a few minutes and have shown to be at least as good as those resulting from the experience of the industrial partner—and a laborious procedure of trials and errors.

GAGERO and its Pareto strategy is now being compared to other zero order optimization tools which reduce the multi-objective aspects of the problems into a single-objective via a weighted combination.

GAGERO has been extended to binary coding and was successfully applied to the problem of *Pump Scheduling*, i.e., finding the best strategy to use a number of pumps supplying water to a variable demand, while minimizing the energy consumption and the number of on–off switchs.

Acknowledgements

This work was funded by the Région Wallonne (RW) and the Fonds Social Europen (FSE) in the framework of a First Europe Program (project 991/4329). The authors would like to thank the industrial partner (Techspace Aero - Snecma Group) for providing the pump performance code and useful data.

References

[1] T. Bäck, F. Hoffmeister, Extended selection mechanisms in genetic algorithms, in: R. Belew, L. Booker (Eds.), Proceedings of the Fourth International Conference on Genetic Algorithms and their Applications, Morgan Kaufmann Publishers, San Mateo, CA, 1991, pp. 92–99.
[2] J. Baker, Adaptive selection methods for genetic algorithms, in: J. Grefenstette (Ed.), Proceedings of the First International Conference on Genetic Algorithm and their Applications, Lawrence Erlbaum Associates, Hillsdale, NJ, 1985, pp. 101–111.
[3] C. Coello, A comprehensive survey of evolutionary-based multiobjective optimization techniques, Knowledge Inform. Systems 1 (3) (1999) 269–308.
[4] T. Dickenson, Pumping Manual, 9th Edition, Elsevier Advanced Technology, 1995, ISBN 1 85617215 5.
[5] L. Eshelman, S. Schaffer, Real-coded genetic algorithms and interval schemata, in: L. Whitley (Ed.), Foundations of Genetic Algorithms, Vol. 2, Morgan Kaufmann Publishers, San Mateo, CA, 1993, pp. 187–202.
[6] C. Fonseca, Multiobjective genetic algorithms with application to control engineering problems, Ph.D. Thesis, University of Sheffield, 1995.
[7] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company Inc., Reading, MA, 1989.
[8] D. Goldberg, K. Deb, A comparative analysis of selection scheme used in genetic algorithms, in: G. Rawlins (Ed.), Foundations of Genetic Algorithms, Morgan Kaufmann Publishers, San Mateo, CA, 1991, pp. 69–93.
[9] J. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor, 1975.
[10] J. Joines, C. Houck, On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs, in: Proceedings of the First IEEE International Conference on Evolutionary Computation, IEEE Press, New York, 1994, pp. 579–584.
[11] V. Kelner, Etude de l'adéquation optimale à un ensemble de spécifications d'un empilement de pompes travaillant en parallèle, Rapport technique Convention RW 991/4329, Université de Liège, Décembre 2001.
[12] V. Pareto, Cours d'économie politique, F. Rouge, Lausanne, 1896.
[13] H. Pohlheim, Geatbx: Genetic and evolutionary algorithm toolbox for use with matlab, http://www.systemtechnik.tu-ilmenau.de/~pohlheim/GA_Toolbox.
[14] C. Poloni, Multi-criteria optimisation, constraint handling with GAs, in: Lecture Series 2000-07: Genetic Algorithms for Optimisation in Aeronautics and Turbomachinery, von Karman Institute, 2000.
[15] Rolls-Royce, The Jet Engine, Lubrication, 1999, pp. 73–83 (Chapter 8).
[16] D. Van Veldhuizen, Multiobjective evolutionary algorithms: classifications, analyses, and new innovations, Ph.D. Thesis, Department of Electrical and Computer Engineering, Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, OH, May 1999.