Chapter 3

Introduction to Evolutionary Multi-objective Optimization

Kalyanmoy Deb

Department of Mechanical Engineering and
Indian Institute of Technology Kanpur
Kanpur, PIN 208016, India
deb@iitk.ac.in
http://www.iitk.ac.in/kangal/deb.htm

Department of Business Technology
Helsinki School of Economics
PO Box 1210, 00101 Helsinki, Finland
Kalyanmoy.Deb@hse.fi

Summary. In its current state, evolutionary multi-objective optimization (EMO) is an established field of research and application with more than 150 PhD theses, more than ten dedicated texts and edited books, commercial softwares and numerous freely downloadable codes, a biannual conference series running successfully since 2001, special sessions and workshops held at all major evolutionary computing conferences, and full-time researchers from universities and industries from all around the globe. In this chapter, we provide a brief introduction to EMO principles, illustrate some EMO algorithms with simulated results, and outline the current research and application potential of EMO. For solving multi-objective optimization problems, EMO procedures attempt to find a set of well-distributed Pareto-optimal points, so that an idea of the extent and shape of the Pareto-optimal front can be obtained. Although this task was the early motivation of EMO research, EMO principles are now being found to be useful in various other problem solving tasks, enabling one to treat problems naturally as they are. One of the major current research thrusts is to combine EMO procedures with other multiple criterion decision making (MCDM) tools so as to develop hybrid and interactive multi-objective optimization algorithms for finding a set of trade-off optimal solutions and then choose a preferred solution for implementation. This chapter provides the background of EMO principles and their potential to launch such collaborative studies with MCDM researchers in the coming years.

3.1 Introduction

In a short span of about fourteen years since the suggestion of the first set of successful algorithms, evolutionary multi-objective optimization (EMO) has now become a popular and useful field of research and application. In a recent survey announced during the World Congress on Computational Intelligence (WCCI) in Vancouver 2006, EMO has been judged as one of the three fastest growing fields of research and application among all computational

intelligence topics. Evolutionary optimization (EO) algorithms use a population based approach in which more than one solution participates in an iteration and evolves a new population of solutions in each iteration. The reasons for their popularity are many. Some of them are: (i) EOs do not require any derivative information (ii) EOs are relatively simple to implement and (iii) EOs are flexible and have a wide-spread applicability. For solving single-objective optimization problems or in other tasks focusing on finding a single optimal solution, the use of a population of solutions in each iteration may at first seem like an overkill (but they help provide an implicit parallel search ability, thereby making EOs computationally efficient (Holland, 1975; Goldberg, 1989)), in solving multi-objective optimization problems an EO procedure is a perfect match (Deb, 2001). The multi-objective optimization problems, by nature, give rise to a set of Pareto-optimal solutions which need a further processing to arrive at a single preferred solution. To achieve the first task, it becomes quite a natural proposition to use an EO, because the use of population in an iteration helps an EO to simultaneously find multiple non-dominated solutions, which portrays a trade-off among objectives, in a single simulation run.

In this chapter, we begin with a brief description of an evolutionary optimization procedure for single-objective optimization. Thereafter, we describe the principles of evolutionary multi-objective optimization and sketch a brief history of how the field has evolved over the past one-and-a-half decades. To generate interest in the minds of the readers, we also provide a description of a few well-known EMO procedures with some representative simulation studies. Finally, we discuss the achievements of EMO research and its current focus. It is clear from these discussions that EMO is not only being found to be useful in solving multi-objective optimization problems, it is also helping to solve other kinds of optimization problems in a better manner than they are traditionally solved. As a by-product, EMO-based solutions are helping to reveal important hidden knowledge about a problem – a matter which is difficult to achieve otherwise.

However, much of the research focus in EMO had been to find a set of near Pareto-optimal solutions and not much studies have been made yet to execute the remaining half of a multi-objective optimization task – selection of a single preferred Pareto-optimal solution for implementation. Although a few studies have just scratched the surface in this direction, this book remains as a major step towards achieving possible EMO and multi-criterion decision-making hybrids. This chapter provides a sketch of the EMO background, some representative EMO procedures, and EMO's achievements and potential, so that readers can get attracted to the EMO literature and engage in collaborative research and application using EMO and MCDM.

3.2 Evolutionary Optimization (EO): A Brief Introduction

Evolutionary optimization principles are different from classical optimization methodologies in the following main ways (Goldberg, 1989):

- An EO procedure does not usually use gradient information in its search process. Thus, EO methodologies are direct search procedures, allowing them to be applied to a wide variety of optimization problems. Due to this reason, EO methodologies may not be competitive with specific gradient-based optimization procedures in solving more structured and well-behaved optimization problems such as convex programming, linear or quadratic programming.
- An EO procedure uses more than one solution (a *population* approach) in an iteration, unlike in most classical optimization algorithms which updates one solution in each iteration (a *point* approach). The use of a population has a number of advantages: (i) it provides an EO with a parallel processing power achieving a computationally quick overall search, (ii) it allows an EO to find multiple optimal solutions, thereby facilitating the solution of multi-modal and multi-objective optimization problems, and (iii) it provides an EO with the ability to normalize decision variables (as well as objective and constraint functions) within an evolving population using the population-best minimum and maximum values. However, the flip side of working with a population of solutions is the computational cost and memory associated with executing each iteration.
- An EO procedure uses stochastic operators, unlike deterministic operators used in most classical optimization methods. The operators tend to achieve a desired effect by using biased probability distributions towards desirable outcomes, as opposed to using predetermined and fixed transition rules. This allows an EO algorithm to negotiate multiple optima and other complexities better and provide them with a global perspective in their search.

An EO begins its search with a population of solutions usually created at random within a specified lower and upper bound on each variable. If bounds are not supplied in an optimization problem, suitable values can be assumed only for the initialization purpose. Thereafter, the EO procedure enters into an iterative operation of updating the current population to create a new population by the use of four main operators: selection, crossover, mutation and elite-preservation. The operation stops when one or more pre-specified termination criteria are met. Thus, an EO takes the following simple structure in which the terms shown in bold (but not underlined) are changeable by the user.

An Evolutionary Optimization Procedure:
begin
  $t = 0$;
  Initialization(P_t);
  do
    Evaluation(P_t);
    $P'_t =$ **Selection**(P_t);
    $P''_t =$ **Variation**(P'_t);
    $P_{t+1} =$ **Elitism**(P_t, P''_t);
    $t = t + 1$; **od**;
  while (**Termination**(P_t, P_{t+1})); **od**
end

The initialization procedure is already described above. If in a problem the knowledge of some good solutions is available, it is wise to use such information in creating the initial population P_0. Elsewhere (Deb *et al.*, 2003a), it is highlighted that for solving complex real-world optimization problems, such a customized initialization is useful and also helpful in achieving a faster search.

The evaluation of a population means computation of each population member for its objective function value, constraint values and determining if the solution is feasible. Since this requires evaluation of multiple functions, this procedure also requires a *relative* preference order (or sorting) of solutions (say from best to worst) in the population. Often, such an ordering can be established by creating a real-valued *fitness* function derived from objective and constraint values. For multi-objective optimization problems, one of the ways to achieve the ordering is to sort the population based on a domination principle Goldberg (1989). It is interesting to note that since an ordering of best-to-worst is enough for the evaluation purpose, EO procedures allow handling of different problem types: dynamically changing problems, problems which are not mathematically expressed, problems which are procedure-based, and others.

After the population members are evaluated, the selection operator chooses above-average (in other words, better) solutions with a larger probability to fill an intermediate mating pool. For this purpose, several stochastic selection operators exist in the EO literature. In its simplest form (called the *tournament* selection (Deb, 1999a)), two solutions can be picked at random from the evaluated population and the better of the two (in terms of its evaluated order) can be picked.

The 'variation' operator is a collection of a number of operators (such as crossover, mutation etc.) which are used to generate a modified population. The purpose of the crossover operator is to pick two or more solutions (parents) randomly from the mating pool and create one or more solutions by exchanging information among the parent solutions. The crossover operator is applied with a crossover probability ($p_c \in [0, 1]$), indicating the proportion of population members participating in the crossover operation. The remaining

$(1 - p_c)$ proportion of the population is simply copied to the modified (child) population. In the context of real-parameter optimization having n real-valued variables and involving a crossover with two parent solutions, each variable may be crossed at a time. A probability distribution which depends on the difference between the two parent variable values is often used to create two new numbers as child values around the two parent values. One way to set the probability distribution is that if the two parent values are quite different, the created child values are also set to different from their parent values, thereby allowing a broader search to take place. This process is usually helpful during the early generations of an EO when population members are quite different from each other. However, after some generations when population members tend to get close to each other to converge to an interesting region in the search space due to the interactions of EO operators, the same crossover operator must then focus the search by creating child values closer to parent values. This varying action of the EO crossover operator without any intervention from the user provides a *self-adaptive* feature to a real-parameter EO algorithm. There exists a number of probability distributions for achieving such a self-adaptive crossover operation (Deb, 2001; Herrera *et al.*, 1998). A unimodal probability distribution with its mode at the parent variable value is used to design the simulated binary crossover (SBX) and is popularly used in the EO literature. Besides the variable-wise recombination operators, vector-wise recombination operators also suggested to propagate the correlation among variables of parent solutions to the created child solutions (Deb *et al.*, 2002a; Storn and Price, 1997).

Each child solution, created by the crossover operator, is then perturbed in its vicinity by a mutation operator (Goldberg, 1989). Every variable is mutated with a mutation probability p_m, usually set as $1/n$ (n is the number of variables), so that on an average one variable gets mutated per solution. In the context of real-parameter optimization, a simple Gaussian probability distribution with a predefined variance can be used with its mean at the child variable value (Deb, 2001). This operator allows an EO to search locally around a solution and is independent on the location of other solutions in the population.

The elitism operator combines the old population with the newly created population and chooses to keep better solutions from the combined population. Such an operation makes sure that an algorithm has a monotonically non-degrading performance. Rudolph (1994) proved an asymptotic convergence of a specific EO but having elitism and mutation as two essential operators.

Finally, the user of an EO needs to choose termination criteria. Often, a predetermined number of generations is used as a termination criterion. For goal attainment problems, an EO can be terminated as soon as a solution with a predefined goal or a target solution is found. In many studies (Goldberg, 1989; Michalewicz, 1992; Gen and Cheng, 1997; Bäck *et al.*, 1997), a termination criterion based on the statistics of the current population vis-a-vis that of the previous population to determine the rate of convergence is used. In other

more recent studies, theoretical optimality conditions (such as the extent of satisfaction of Karush-Kuhn-Tucker (KKT) conditions) are used to determine the termination of a real-parameter EO algorithm (Deb *et al.*, 2007). Although EOs are heuristic based, the use of such theoretical optimality concepts in an EO can also be used to test their converging abilities towards local optimal solutions.

Thus, overall an EO procedure is a population-based stochastic search procedure which iteratively emphasizes its better population members, uses them to recombine and perturb locally in the hope of creating new and better populations until a predefined termination criterion is met. The use of a population helps to achieve an *implicit parallelism* (Goldberg, 1989; Holland, 1975; Vose *et al.*, 2003) in an EO's search mechanism (an inherent parallel search in different regions of the search space), a matter which makes an EO computationally attractive for solving difficult problems. In the context of certain Boolean functions, a computational time saving to find the optimum varying polynomial to the population size is proven (Jansen and Wegener, 2001). On one hand, the EO procedure is flexible, thereby allowing a user to choose suitable operators and problem-specific information to suit a specific problem. On the other hand, the flexibility comes with an onus on the part of a user to choose appropriate and tangible operators so as to create an efficient and consistent search (Radcliffe, 1991). However, the benefits of having a flexible optimization procedure, over their more rigid and specific optimization algorithms, provide hope in solving difficult real-world optimization problems involving non-differentiable objectives and constraints, non-linearities, discreteness, multiple optima, large problem sizes, uncertainties in computation of objectives and constraints, uncertainties in decision variables, mixed type of variables, and others. A wiser approach to solving optimization problems of the real world would be to first understand the niche of both EO and classical methodologies and then adopt hybrid procedures employing the better of the two as the search progresses over varying degrees of search-space complexity from start to finish.

3.2.1 EO Terminologies

Evolutionary optimization literature uses somewhat different terminologies than those used in classical optimization. The following brief descriptions may be beneficial for the readers from other optimization fields:

Children: New solutions (or decision variable vectors) created by a combined effect of crossover and mutation operators.

Crossover: An operator in which two or more parent solutions are used to create (through recombination) one or more child solutions.

Crossover probability: The probability of performing a crossover operation. This means, on average, the proportion of population members participating in crossover operation in a generation.

Differential evolution (DE): A particular evolutionary optimization procedure which is usually applied to solve real-parameter optimization problems. A good reference on DE is available from Price *et al.* (2005).

Distribution index: A non-negative parameter (η_c) used for implementing the real-parameter simulated binary crossover (SBX). For achieving a mutation operation in the real-valued search space, a polynomial probability distribution with its mode at the parent variable value and with monotonically reducing probability for creating child values away from parent was suggested (Deb, 2001). This operator involves a non-negative parameter (η_m), which must also be set the user. In both the above recombination and mutation operations, a large distribution index value means a probability distribution with a small variance. For single-objective EO, $\eta_c = 2$ is commonly used and for multi-objective EAs, $\eta_c \in [5, 10]$ is used, whereas $\eta_m \approx 20$ is usually used in both cases. A study (Deb and Agrawal, 1999) has shown that with η_m, a perturbation to the order of $O(\eta_m^{-1})$ takes place to the mutated variable from its parent solution to the mutated solution.

Elitism: An operator which preserves the better of parent and child solutions (or populations) so that a previously found better solution is never deleted.

Evolutionary algorithm (EA): A generic name given to an algorithm which applies Darwinian survival-of-the-fittest evolutionary principles along with genetically motivated recombination and mutation principles in a stochastic manner usually to a population of solutions to iteratively create a new and hopefully better population of solutions in the context of a stationary or a dynamic fitness landscape.

Evolutionary optimization (EO): An EA which is designed to solve an optimization problem.

Evolutionary programming (EP): An EA originally applied to a set of finite-state machines to evolve better learning machines (Fogel *et al.*, 1966). Now, EPs are used for various optimization tasks including real-parameter optimization and are somewhat similar to evolution strategies. EP mainly depends on its selection and mutation operators.

Evolution strategy (ES): An EA which is mostly applied to real-valued decision variables and is mainly driven by a selection and a mutation operator, originally suggested by P. Bienert, I. Rechenberg and H.-P. Schwefel of the Technical University of Berlin. Early applications were experimental based (Rechenberg, 1965; Schwefel, 1968) and some texts (Rechenberg, 1973; Schwefel, 1995) provide details about ES procedures.

Fitness: A fitness or a fitness landscape is a function derived from objective function(s), constraint(s) and other problem descriptions which is used in the selection (or reproduction) operator of an EA. A solution is usually called better than the other, if its fitness function value is better.

Generation: An iteration of an EA.

Generational EA: An EA in which a complete set of child solutions (equal to the population size N) is first created by EA operators (usually, selection, crossover and mutation) before comparing with parent solutions to

decide which solutions qualify as members of the new population. Thus, in one iteration, a complete population of N members replaces the old population.

Genetic algorithm (GA): An early version of an EA, originally proposed by Holland (1962, 1975), which uses three main operators – selection, crossover and mutation – on a population of solutions at every generation. In binary-coded GAs, solutions are represented in a string of binary digits (bits). In real-parameter GAs, solutions are represented as a vector of real-parameter decision variables. Other representations can also be used to suit the handling of a problem.

Genetic programming (GP): An EA which works on computer programs (usually C or Lisp codes) or on graphs, trees, etc., representing a solution methodology to mainly evolve optimal learning strategies (Koza, 1992).

Individual: An EA population member representing a solution to the problem at hand.

Mating Pool: An intermediate population (usually created by the selection operator) used for creating new solutions by crossover and mutation operators.

Mutation: An EA operator which is applied to a single solution to create a new perturbed solution. A fundamental difference with a crossover operator is that mutation is applied to a single solution, whereas crossover is applied to more than one solution.

Mutation probability: The probability of performing a mutation operation. This refers to, on average, the proportion of decision variables participating in a mutation operation to a solution.

Niching: A niching is an operator by which selection pressure of population members are controlled so as to not allow a single solution to take over the population. Thus, niching helps to maintain a diverse population. A number of niching techniques exist, but the *sharing* approach (Goldberg and Richardson, 1987) is popularly used to find a set of multiple optimal solutions in a multi-modal optimization problem.

Offspring: Same as Children, defined above.

Parent: A solution used during crossover operation to create a child solution.

Particle swarm optimization (PSO): An EA which updates each population member by using a weighted sum of two concepts borrowed from the movement of natural swarm of birds (or fishes etc.) – inclination to move towards its own individual best position and towards the swarm's best position in the decision variable space. A good source for more information is (Kennedy *et al.*, 2001).

Population: A set of solutions used in one generation of an EA. The number of solutions in a population is called 'population size'.

Recombination: Same as crossover, defined above.

Reproduction: An EA operator which mimics Darwin's survival of the fittest principle by making duplicate copies of above-average solutions in the population at the expense of deleting below-average solutions. Initial EA stud-

ies used a *proportionate* reproduction procedure in which multiple copies of a population member are assigned to the mating pool proportionate to the individual's fitness. Thus, this operator is used for maximization problems and for fitness values which are non-negative. Current studies use tournament selection which compares two population members based on their fitness values and sends the better solution to the mating pool. This operator does not have any limitation on fitness function.

Selection: Same as Reproduction, defined above.

Selection pressure: The extent of emphasis given to above-average solution in a selection operator. No real quantifiable definition exists, but loosely it is considered as the number of copies allocated to the population-best solution by the selection operator Goldberg *et al.* (1993).

Sharing strategy: It is niching operation in which each population member's fitness is divided by a niche-count (in some sense, a niche-count is an estimate of number of other population members around an individual) and a shared fitness is computed. A proportionate selection procedure is then used with the shared fitness values to create the mating pool.

Solution: An EA population member, same as an 'individual'.

Steady-state EA: An EA in which only one new population member is added to the population in one generation.

String: In a binary-coded GA, a population member, made of a collection of bits, is called a string.

3.3 Evolutionary Multi-objective Optimization (EMO)

A multi-objective optimization problem involves a number of objective functions which are to be either minimized or maximized. As in a single-objective optimization problem, the multi-objective optimization problem may contain a number of constraints which any feasible solution (including all optimal solutions) must satisfy. Since objectives can be either minimized or maximized, we state the multi-objective optimization problem in its general form:

$$\left.\begin{array}{rll} \text{Minimize/Maximize } f_m(\mathbf{x}), & & m = 1, 2, \ldots, M; \\ \text{subject to } g_j(\mathbf{x}) \geq 0, & & j = 1, 2, \ldots, J; \\ h_k(\mathbf{x}) = 0, & & k = 1, 2, \ldots, K; \\ x_i^{(L)} \leq x_i \leq x_i^{(U)}, & & i = 1, 2, \ldots, n. \end{array}\right\} \tag{3.1}$$

A solution $\mathbf{x} \in \mathbf{R}^n$ is a vector of n decision variables: $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$. The solutions satisfying the constraints and variable bounds constitute a *feasible decision variable space* $S \subset \mathbf{R}^n$. One of the striking differences between single-objective and multi-objective optimization is that in multi-objective optimization the objective functions constitute a multi-dimensional space, in addition to the usual decision variable space. This additional M-dimensional space is called the *objective space*, $Z \subset \mathbf{R}^M$. For each solution $\mathbf{x}$ in the decision

variable space, there exists a point $\mathbf{z} \in \mathbf{R}^M$) in the objective space, denoted by $\mathbf{f}(\mathbf{x}) = \mathbf{z} = (z_1, z_2, \ldots, z_M)^T$. To make the descriptions clear, we refer a 'solution' as a variable vector and a 'point' as the corresponding objective vector.

The optimal solutions in multi-objective optimization can be defined from a mathematical concept of *partial ordering*. In the parlance of multi-objective optimization, the term *domination* is used for this purpose. In this section, we restrict ourselves to discuss unconstrained (without any equality, inequality or bound constraints) optimization problems. The domination between two solutions is defined as follows (Deb, 2001; Miettinen, 1999):

Definition 1. *A solution $\mathbf{x}^{(1)}$ is said to dominate the other solution $\mathbf{x}^{(2)}$, if both the following conditions are true:*

1. *The solution $\mathbf{x}^{(1)}$ is no worse than $\mathbf{x}^{(2)}$ in all objectives. Thus, the solutions are compared based on their objective function values (or location of the corresponding points ($\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$) on the objective space).*
2. *The solution $\mathbf{x}^{(1)}$ is strictly better than $\mathbf{x}^{(2)}$ in at least one objective.*

For a given set of solutions (or corresponding points on the objective space, for example, those shown in Figure 3.1(a)), a pair-wise comparison can be made using the above definition and whether one point dominates the other can be established. All points which are not dominated by any other member

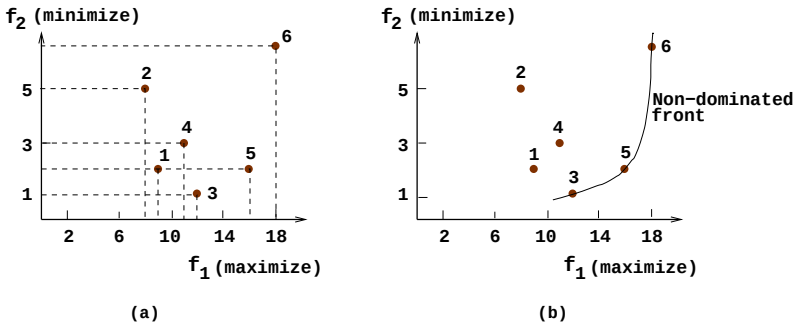


Fig. 3.1. A set of points and the first non-domination front are shown.

of the set are called the non-dominated points of class one, or simply the non-dominated points. For the set of six solutions shown in the figure, they are points 3, 5, and 6. One property of any two such points is that a gain in an objective from one point to the other happens only due to a sacrifice in at least one other objective. This *trade-off* property between the non-dominated points makes the practitioners interested in finding a wide variety of them before making a final choice. These points make up a front when viewed them together on the objective space; hence the non-dominated points

are often visualized to represent a *non-domination front*. The computational effort needed to select the points of the non-domination front from a set of N points is $O(N \log N)$ for 2 and 3 objectives, and $O(N \log^{M-2} N)$ for $M > 3$ objectives (Kung *et al.*, 1975).

With the above concept, now it is easier to define the *Pareto-optimal solutions* in a multi-objective optimization problem. If the given set of points for the above task contain all points in the search space (assuming a countable number), the points lying on the non-domination front, by definition, do not get dominated by any other point in the objective space, hence are Pareto-optimal points (together they constitute the Pareto-optimal front) and the corresponding pre-images (decision variable vectors) are called Pareto-optimal solutions. However, more mathematically elegant definitions of Pareto-optimality (including the ones for continuous search space problems) exist in the multi-objective literature and are discussed in Chapters **??** and **??**.

Similar to local optimal solutions in single objective optimization, local Pareto-optimal solutions are also defined in multi-objective optimization (Deb, 2001; Miettinen, 1999):

Definition 2. *If for every member* $\mathbf{x}$ *in a set* $\underline{P}$ *there exists no solution* $\mathbf{y}$ *(in the neighborhood of* $\mathbf{x}$ *such that* $\|\mathbf{y} - \mathbf{x}\|_\infty \leq \epsilon$*, where* ϵ *is a small positive scalar) dominating any member of the set* $\underline{P}$*, then solutions belonging to the set* $\underline{P}$ *constitute a local Pareto-optimal set.*

3.3.1 EMO Principles

In the context of multi-objective optimization, the extremist principle of finding the optimum solution cannot be applied to one objective alone, when the rest of the objectives are also important. Different solutions may produce trade-offs (conflicting outcomes among objectives) among different objectives. A solution that is extreme (in a better sense) with respect to one objective requires a compromise in other objectives. This prohibits one to choose a solution which is optimal with respect to only one objective. This clearly suggests two ideal goals of multi-objective optimization:

1. Find a set of solutions which lie on the Pareto-optimal front, and
2. Find a set of solutions which are diverse enough to represent the entire range of the Pareto-optimal front.

Evolutionary multi-objective optimization (EMO) algorithms attempt to follow both the above principles similar to the other a posteriori MCDM methods (refer to Chapter **??**).

Although one fundamental difference between single and multiple objective optimization lies in the cardinality in the optimal set, from a practical standpoint a user needs only one solution, no matter whether the associated optimization problem is single or multi-objective. The user is now in

a dilemma. Since a number of solutions are optimal, the obvious question arises: Which of these optimal solutions must one choose? This is not an easy question to answer. It involves higher-level information which is often non-technical, qualitative and experience-driven. However, if a set of many trade-off solutions are already worked out or available, one can evaluate the pros and cons of each of these solutions based on all such non-technical and qualitative, yet still important, considerations and compare them to make a choice. Thus, in a multi-objective optimization, ideally the effort must be made in finding the set of trade-off optimal solutions by considering all objectives to be important. After a set of such trade-off solutions are found, a user can then use higher-level qualitative considerations to make a choice. Since an EMO procedure deals with a population of solutions in every iteration, it makes them intuitive to be applied in multi-objective optimization to find a set of non-dominated solutions. Like other a posteriori MCDM methodologies, an EMO based procedure works with the following principle in handling multi-objective optimization problems:

Step 1 Find multiple non-dominated points as close to the Pareto-optimal front as possible, with a wide trade-off among objectives.
Step 2 Choose one of the obtained points using higher-level information.

Figure 3.2 shows schematically the principles, followed in an EMO procedure. Since EMO procedures are heuristic based, they may not guarantee in finding Pareto-optimal points, as a theoretically provable optimization method would do for tractable (for example, linear or convex) problems. But EMO procedures have essential operators to constantly improve the evolving non-dominated points (from the point of view of convergence and diversity

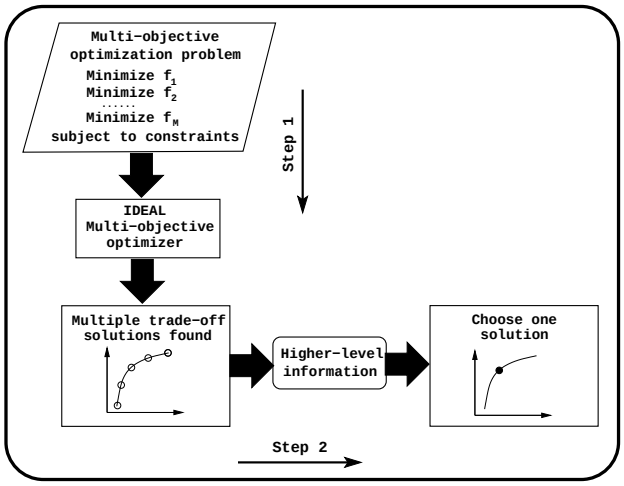


Fig. 3.2. Schematic of a two-step multi-objective optimization procedure.

discussed above) similar to the way most natural and artificial evolving systems continuously improve their solutions. To this effect, a recent simulation study (Deb *et al.*, 2007) has demonstrated that a particular EMO procedure, starting from random non-optimal solutions, can progress towards theoretical Karush-Kuhn-Tucker (KKT) points with iterations in real-valued multi-objective optimization problems. The main difference and advantage of using an EMO compared to a posteriori MCDM procedures is that multiple trade-off solutions can be found in a single simulation run, as most a posteriori MCDM methodologies would require multiple applications.

In Step 1 of the EMO-based multi-objective optimization (the task shown vertically downwards in Figure 3.2), multiple trade-off, non-dominated points are found. Thereafter, in Step 2 (the task shown horizontally, towards the right), higher-level information is used to choose one of the obtained trade-off points. This dual task allows an interesting feature, if applied for solving single-objective optimization problems. It is easy to realize that a single-objective optimization is a degenerate case of multi-objective optimization, as shown in details in another study (Deb and Tiwari, 2008). In the case of single-objective optimization having only one globally optimal solution, Step 1 will ideally find only one solution, thereby not requiring us to proceed to Step 2. However, in the case of single-objective optimization having multiple global optima, both steps are necessary to first find all or multiple global optima, and then to choose one solution from them by using a higher-level information about the problem. Thus, although seems ideal for multi-objective optimization, the framework suggested in Figure 3.2 can be ideally thought as a generic principle for both single and multiple objective optimization.

3.3.2 A Posteriori MCDM Methods and EMO

In the 'a posteriori' MCDM approaches (also known as 'generating MCDM methods'), the task of finding multiple Pareto-optimal solutions is achieved by executing many independent single-objective optimizations, each time finding a single Pareto-optimal solution (see Miettinen (1999) and Chapter **??** of this book). A parametric scalarizing approach (such as the weighted-sum approach, ϵ-constraint approach, and others) can be used to convert multiple objectives into a parametric single-objective objective function. By simply varying the parameters (weight vector or ϵ-vector) and optimizing the scalarized function, different Pareto-optimal solutions can be found. In contrast, in an EMO, multiple Pareto-optimal solutions are attempted to be found in a single simulation by emphasizing multiple non-dominated and isolated solutions. We discuss a little later some EMO algorithms describing how such dual emphasis is provided, but now discuss qualitatively the difference between a posteriori MCDM and EMO approaches.

Consider Figure 3.3, in which we sketch how multiple independent parametric single-objective optimizations may find different Pareto-optimal solutions. The Pareto-optimal front corresponds to global optimal solutions of

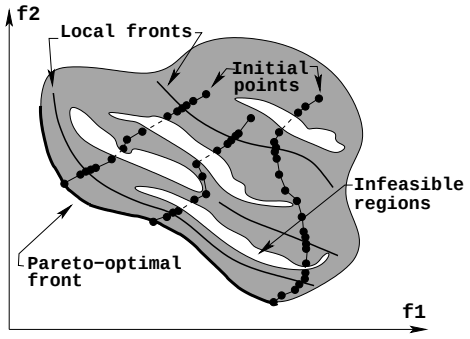


Fig. 3.3. Posteriori MCDM methodology employs independent single-objective optimizations.

several scalarized objectives. However, during the course of an optimization task, algorithms must overcome a number of difficulties, such as infeasible regions, local optimal solutions, flat regions of objective functions, isolation of optimum, etc., to converge to the global optimal solution. Moreover, due to practical limitations, an optimization task must also be completed in a reasonable computational time. This requires an algorithm to strike a good balance between the extent of these tasks its search operators must do to overcome the above-mentioned difficulties reliably and quickly. When multiple simulations are to performed to find a set of Pareto-optimal solutions, the above balancing act must have to performed in every single simulation. Since simulations are performed independently, no information about the success or failure of previous simulations is used to speed up the process. In difficult multi-objective optimization problems, such memory-less a posteriori methods may demand a large overall computational overhead to get a set of Pareto-optimal solutions. Moreover, even though the convergence can be achieved in some problems, independent simulations can never guarantee finding a good distribution among obtained points.

EMO, as mentioned earlier, constitutes an inherent parallel search. When a population member overcomes certain difficulties and make a progress towards the Pareto-optimal front, its variable values and their combination reflect this fact. When a recombination takes place between this solution and other population members, such valuable information of variable value combinations gets shared through variable exchanges and blending, thereby making the overall task of finding multiple trade-off solutions a parallelly processed task.

3.4 History of EMO and Non-Elitist Methodologies

During the early years, EA researchers have realized the need of solving multi-objective optimization problems in practice and mainly resorted to using

weighted-sum approaches to convert multiple objectives into a single goal (Rosenberg, 1967; Fogel *et al.*, 1966).

However, the first implementation of a real multi-objective evolutionary algorithm (vector-evaluated GA or VEGA) was suggested by David Schaffer in the year 1984 (Schaffer, 1984). Schaffer modified the simple three-operator genetic algorithm (with selection, crossover, and mutation) by performing independent selection cycles according to each objective. The selection method is repeated for each individual objective to fill up a portion of the mating pool. Then the entire population is thoroughly shuffled to apply crossover and mutation operators. This is performed to achieve the mating of individuals of different subpopulation groups. The algorithm worked efficiently for some generations but in some cases suffered from its bias towards some individuals or regions (mostly individual objective champions). This does not fulfill the second goal of EMO, discussed earlier.

Ironically, no significant study was performed for almost a decade after the pioneering work of Schaffer, until a revolutionary 10-line sketch of a new non-dominated sorting procedure suggested by David E. Goldberg in his seminal book on GAs (Goldberg, 1989). Since an EA needs a fitness function for reproduction, the trick was to find a single metric from a number of objective functions. Goldberg's suggestion was to use the concept of *domination* to assign more copies to non-dominated individuals in a population. Since diversity is the other concern, he also suggested the use of a *niching* strategy (Goldberg and Richardson, 1987) among solutions of a non-dominated class. Getting this clue, at least three independent groups of researchers developed different versions of multi-objective evolutionary algorithms during 1993-94. Basically, these algorithms differ in the way a fitness assignment scheme is introduced to each individual. We discuss them briefly in the following paragraphs.

Fonseca and Fleming (1993) suggested a multi-objective GA (MOGA), in which all non-dominated population members are assigned a rank one. Other individuals are ranked by calculating how many solutions (say k) dominated a particular solution. That solution is then assigned a rank $(k+1)$. The selection procedure then chooses lower rank solutions to form the mating pool. Since the fitness of a population member is the same as its rank, many population members will have an identical fitness. MOGA applies a niching technique on solutions having identical fitness to maintain a diverse population. But instead of performing niching on the parameter values, they suggested niching on objective function values. The ranking of individuals according to their non-dominance in the population is an important aspect of the work.

Horn *et al.* (1994) used *Pareto domination tournaments* in their niched-Pareto GA (NPGA). In this method, a *comparison set* comprising of a specific number (t_{dom}) of individuals is picked at random from the population at the beginning of each selection process. Two random individuals are picked from the population for selecting a winner according to the following procedure. Both individuals are compared with the members of the comparison set for

domination. If one of them is non-dominated and the other is dominated, then the non-dominated point is selected. On the other hand, if both are either non-dominated or dominated, a *niche-count* is calculated by simply counting the number of points in the entire population within a certain distance (σ_{share}) in the variable space from an individual. The individual with least niche-count is selected. The proposers of this algorithm has reported that the outcome of the algorithm depends on the chosen value of t_{dom}. Nevertheless, the concept of niche formation among the non-dominated points using the tournament selection is an important aspect of the work.

Srinivas and Deb (1994) developed a non-dominated sorting GA (NSGA) which differs from MOGA in two ways: fitness assignment and the way niching is performed. After the population members belonging to the first non-domination class are identified, they are assigned a dummy fitness value equal to N (population size). A sharing strategy is then used on parameter values (instead of objective function values) to find the niche-count for each individual of the best class. Here, the parameter niche-count of an individual is estimated by calculating *sharing function* values (Goldberg and Richardson, 1987). Interested readers may refer to the original study or (Deb, 2001). For each individual, a shared fitness is then found by dividing the assigned fitness N by the niche-count. Thereafter, the second class of non-dominated solutions (obtained by temporarily discounting solutions of first non-domination class and then finding new non-dominated points) is assigned a dummy fitness value smaller than the least shared fitness of solutions of the previous non-domination class. This process is continued till all solutions are assigned a fitness value. This fitness assignment procedure ensured two matters: (i) a dominated solution is assigned a smaller shared fitness value than any solution which dominated it and (ii) in each non-domination class an adequate diversity is ensured. On a number of test problems and real-world optimization problems, NSGA has been found to provide a wide-spread Pareto-optimal or near Pareto-optimal solutions. However, one difficulty of NSGA is to choose an appropriate niching parameter, which directly affects the maximum distance between two neighboring solutions obtained by NSGA. Although most studies used a fixed value of the niching parameter, an adaptive niche-sizing strategy has been suggested elsewhere (Fonseca and Fleming, 1993).

3.5 Elitist Methodologies

The above-mentioned non-elitist EMO methodologies gave a good head-start to the research and application of EMO, but suffered from the fact that they did not use an elite-preservation mechanism in their procedures. It is mentioned before that an addition of elitism in an EO provides a monotonically non-degrading performance. The second generation EMO algorithms implemented an elite-preserving operator in different ways and gave birth to elitist EMO procedures, such as NSGA-II (Deb *et al.*, 2002b), Strength Pareto EA

(SPEA) (Zitzler and Thiele, 1999), Pareto-archived ES (PAES) (Knowles and Corne, 2000), and others. Since these EMO algorithms are state-of-the-art procedures, we describe one of these algorithms in detail and briefly discuss two other procedures commonly used in EMO studies.

3.5.1 Elitist Non-dominated Sorting GA or NSGA-II

The NSGA-II procedure (Deb *et al.*, 2002b) is one of the popularly used EMO procedures which attempt to find multiple Pareto-optimal solutions in a multi-objective optimization problem and has the following three features:

1. It uses an elitist principle,
2. it uses an explicit diversity preserving mechanism, and
3. it emphasizes non-dominated solutions.

At any generation t, the offspring population (say, Q_t) is first created by using the parent population (say, P_t) and the usual genetic operators. Thereafter, the two populations are combined together to form a new population (say, R_t) of size $2N$. Then, the population R_t classified into different non-domination classes. Thereafter, the new population is filled by points of different non-domination fronts, one at a time. The filling starts with the first non-domination front (of class one) and continues with points of the second non-domination front, and so on. Since the overall population size of R_t is $2N$, not all fronts can be accommodated in N slots available for the new population. All fronts which could not be accommodated are deleted. When the last allowed front is being considered, there may exist more points in the front than the remaining slots in the new population. This scenario is illustrated in Figure 3.4. Instead of arbitrarily discarding some members from the last front, the points which will make the diversity of the selected points the highest are chosen.

The crowded-sorting of the points of the last front which could not be accommodated fully is achieved in the descending order of their *crowding*

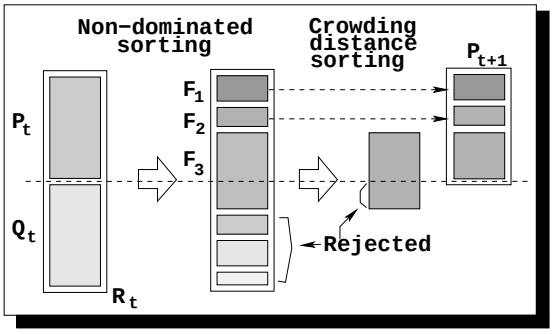


Fig. 3.4. Schematic of the NSGA-II procedure.

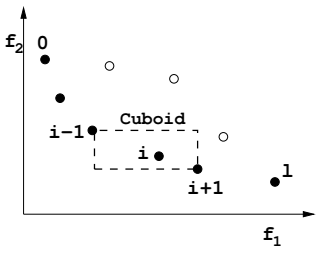


Fig. 3.5. The crowding distance calculation.

distance values and points from the top of the ordered list are chosen. The crowding distance d_i of point i is a measure of the objective space around i which is not occupied by any other solution in the population. Here, we simply calculate this quantity d_i by estimating the perimeter of the cuboid (Figure 3.5) formed by using the nearest neighbors in the objective space as the vertices (we call this the *crowding distance*).

Sample Simulation Results:

Here, we show simulation results of NSGA-II on two test problems. The first problem (ZDT2) is two-objective, 30-variable problem with a concave Pareto-optimal front:

$$\mathsf{ZDT2}: \begin{cases} \text{Minimize } f_1(\mathbf{x}) = x_1, \\ \text{Minimize } f_2(\mathbf{x}) = g(\mathbf{x})\left[1-(f_1(\mathbf{x})/g(\mathbf{x}))^2\right], \\ \text{where} \quad g(\mathbf{x}) = 1+\frac{9}{29}\sum_{i=2}^{30} x_i \\ \qquad\quad 0 \le x_1 \le 1, \\ \qquad\quad -1 \le x_i \le 1, \quad i = 2,3,\ldots,30. \end{cases} \tag{3.2}$$

The second problem (KUR), with three variables, has a disconnected Pareto-optimal front:

$$\mathsf{KUR}: \begin{cases} \text{Minimize } f_1(\mathbf{x}) = \sum_{i=1}^{2}\left[-10\exp(-0.2\sqrt{x_i^2+x_{i+1}^2})\right], \\ \text{Minimize } f_2(\mathbf{x}) = \sum_{i=1}^{3}\left[x_i^{0.8}+5\sin(x_i^3)\right], \\ \qquad\quad -5 \le x_i \le 5, \quad i = 1,2,3. \end{cases} \tag{3.3}$$

NSGA-II is run with a population size of 100 and for 250 generations. The variables are used as real numbers and an SBX recombination operator with $p_c = 0.9$ and distribution index of $\eta_c = 10$ and a polynomial mutation operator (Deb, 2001) with $p_m = 1/n$ (n is the number of variables) and distribution index of $\eta_m = 20$ are used. Figures 3.6 and 3.7 show that NSGA-II converges on the Pareto-optimal front and maintains a good spread of solutions on both test problems.

3.5.2 Strength Pareto EA (SPEA) and SPEA2

Zitzler and Thiele (1998) suggested an elitist multi-criterion EA with the concept of non-domination in their strength Pareto EA (SPEA). They suggested maintaining an external population in every generation storing all non-dominated solutions discovered so far, beginning from the initial population. This external population participates in genetic operations. In each generation, a combined population with the external and the current population is first constructed. All non-dominated solutions in the combined population are assigned a fitness based on the number of solutions they dominate and all dominated solutions are assigned a fitness equal to one more than the sum

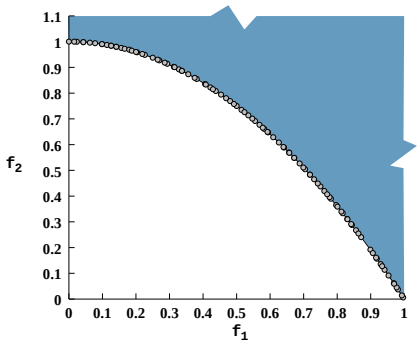


Fig. 3.6. NSGA-II on ZDT2.

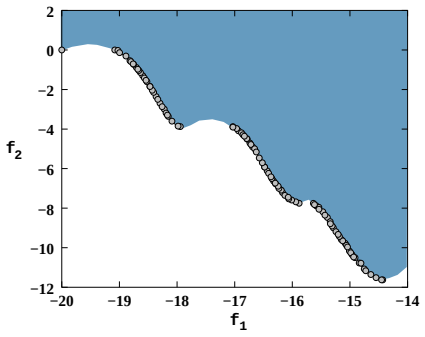


Fig. 3.7. NSGA-II on KUR.

of fitness of solutions which dominate it. This assignment of fitness makes sure that the search is directed towards the non-dominated solutions and simultaneously diversity among dominated and non-dominated solutions are maintained. Diversity is maintained by performing a clustering procedure to maintain a fixed size archive. On knapsack problems, they have reported better results than other methods used in that study.

In their subsequent improved version (SPEA2) (Zitzler *et al.*, 2001b), three changes have been made. First, the archive size is always kept fixed by adding dominated solutions from the EA population, if needed. Second, the fitness assignment procedure for the dominated solutions is slightly different and a density information is used to resolve ties between solutions having identical fitness values. Third, a modified clustering algorithm is used from the k-th nearest neighbor distance estimates for each cluster and special attention is made to preserve the boundary elements.

3.5.3 Pareto Archived ES (PAES) and Pareto Envelope based Selection Algorithms (PESA and PESA2)

Knowles and Corne (2000) suggested Pareto-archived ES (PAES) with one parent and one child. The child is compared with the parent. If the child dominates the parent, the child is accepted as the next parent and the iteration continues. On the other hand, if the parent dominates the child, the child is discarded and a new mutated solution (a new child) is found. However, if the child and the parent do not dominate each other, the choice between child or a parent is resolved by using a crowding procedure. To maintain diversity, an archive of non-dominated solutions found so far is maintained. The child is compared with the archive to check if it dominates any member of the archive. If yes, the child is accepted as the new parent and the dominated solution is eliminated from the archive. If the child does not dominate any member of the archive, both parent and child are checked for their proximity (in terms

of Euclidean distance in the objective space) to the archive members. If the child resides in the least crowded area in the objective space compared to other archive members, it is accepted as a parent and a copy is added to the archive. Later, they suggested a multi-parent PAES with similar principles as above, but applied with a multi-parent evolution strategy framework (Schwefel, 1995). In their subsequent version, called the Pareto Envelope based Selection Algorithm (PESA) (Corne *et al.*, 2000), they combined good aspects of SPEA and PAES. Like SPEA, PESA carries two populations (a smaller EA population and a larger archive population). Non-domination and the PAES crowding concept is used to update the archive with the newly created child solutions.

In an extended version of PESA (or PESA2) (Corne *et al.*, 2001), instead of applying the selection procedure on population members, hyperboxes in the objective space are selected based on the number of solutions residing in the hyperboxes. After hyperboxes are selected, a random solution from the chosen hyperboxes is kept. This region-based selection procedure has shown to perform better than the individual-based selection procedure of PESA. In some sense, the PESA2 selection scheme is similar in concept to ϵ-dominance (Laumanns *et al.*, 2002) in which predefined ϵ values determine the hyperbox dimensions. Other ϵ-dominance based EMO procedures (Deb *et al.*, 2003b) have shown computationally faster and better distributed solutions than NSGA-II or SPEA2.

There also exist other competent EMOs, such as multi-objective messy GA (MOMGA) (Veldhuizen and Lamont, 2000), multi-objective micro-GA (Coello and Toscano, 2000), neighborhood constraint GA (Loughlin and Ranjithan, 1997), ARMOGA (Sasaki *et al.*, 2001), and others. Besides, there exists other EA based methodologies, such as particle swarm EMO (Coello and Lechuga, 2002; Mostaghim and Teich, 2003), ant-based EMO (McMullen, 2001; Gravel *et al.*, 2002), and differential evolution based EMO (Babu and Jehan, 2003).

3.6 Applications of EMO

Since the early development of EMO algorithms in 1993, they have been applied to many real-world and interesting optimization problems. Descriptions of some of these studies can be found in books (Deb, 2001; Coello *et al.*, 2002; Osyczka, 2002), dedicated conference proceedings (Zitzler *et al.*, 2001a; Fonseca *et al.*, 2003; Coello *et al.*, 2005; Obayashi *et al.*, 2007), and domain-specific books, journals and proceedings. In this section, we describe one case study which clearly demonstrates the EMO philosophy which we described in Section 3.3.1.

3.6.1 Spacecraft Trajectory Design

Coverstone-Carroll *et al.* (2000) proposed a multi-objective optimization technique using the original non-dominated sorting algorithm (NSGA) (Srinivas

and Deb, 1994) to find multiple trade-off solutions in a spacecraft trajectory optimization problem. To evaluate a solution (trajectory), the SEPTOP (Solar Electric Propulsion Trajectory Optimization) software (Sauer, 1973) is called for, and the delivered payload mass and the total time of flight are calculated. The multi-objective optimization problem has eight decision variables controlling the trajectory, three objective functions: (i) maximize the delivered payload at destination, (ii) maximize the negative of the time of flight, and (iii) maximize the total number of heliocentric revolutions in the trajectory, and three constraints limiting the SEPTOP convergence error and minimum and maximum bounds on heliocentric revolutions.

On the Earth–Mars rendezvous mission, the study found interesting trade-off solutions (Coverstone-Carroll *et al.*, 2000). Using a population of size 150, the NSGA was run for 30 generations. The obtained non-dominated solutions are shown in Figure 3.8 for two of the three objectives and some selected solutions are shown in Figure 3.9. It is clear that there exist short-time flights

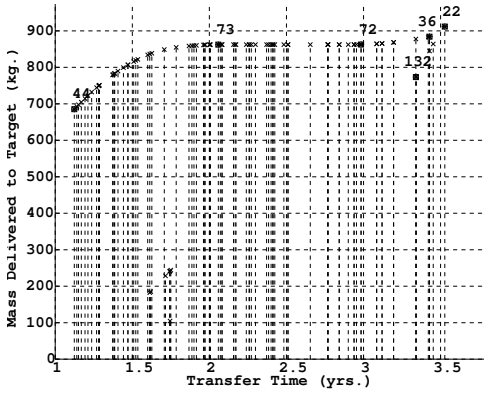


Fig. 3.8. Obtained non-dominated solutions using NSGA.

with smaller delivered payloads (solution marked 44) and long-time flights with larger delivered payloads (solution marked 36). Solution 44 can deliver a mass of 685.28 kg and requires about 1.12 years. On other hand, an intermediate solution 72 can deliver almost 862 kg with a travel time of about 3 years. In these figures, each continuous part of a trajectory represents a *thrusting* arc and each dashed part of a trajectory represents a *coasting* arc. It is interesting to note that only a small improvement in delivered mass occurs in the solutions between 73 and 72 with a sacrifice in flight time of about an year.

The multiplicity in trade-off solutions, as depicted in Figure 3.9, is what we envisaged in discovering in a multi-objective optimization problem by using a posteriori procedure, such as an EMO algorithm. This aspect was also discussed in Figure 3.2. Once such a set of solutions with a good trade-off

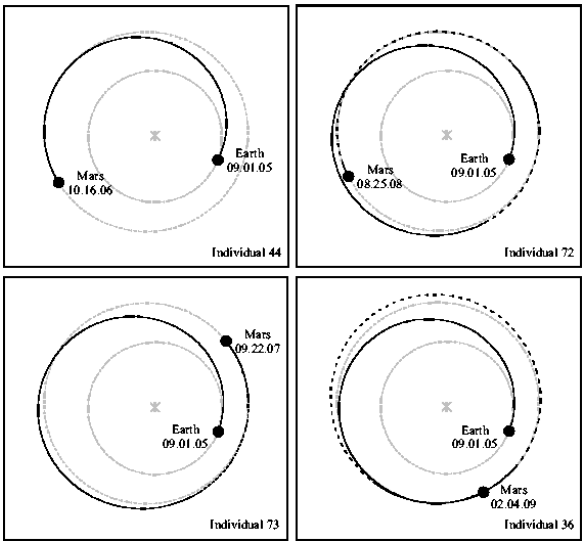


Fig. 3.9. Four trade-off trajectories.

among objectives is obtained, one can analyze them for choosing a particular solution. For example, in this problem context, it makes sense to not choose a solution between points 73 and 72 due to poor trade-off between the objectives in this range. On the other hand, choosing a solution within points 44 and 73 is worthwhile, but which particular solution to choose depends on other mission related issues. But by first finding a wide range of possible solutions and revealing the shape of front, EMO can help narrow down the choices and allow a decision maker to make a better decision. Without the knowledge of such a wide variety of trade-off solutions, a proper decision-making may be a difficult task. Although one can choose a scalarized objective (such as the ϵ-constraint method with a particular ϵ vector) and find the resulting optimal solution, the decision-maker will always wonder what solution would have been derived if a different ϵ vector was chosen. For example, if $\epsilon_1 = 2.5$ years is chosen and mass delivered to the target is maximized, a solution in between points 73 and 72 will be found. As discussed earlier, this part of the Pareto-optimal front does not provide the best trade-offs between objectives that this problem can offer. A lack of knowledge of good trade-off regions before a decision is made may allow the decision maker to settle for a solution which, although optimal, may not be a good compromised solution. The EMO procedure allows a flexible and a pragmatic procedure for finding a well-diversified set of solutions simultaneously so as to enable picking a particular region for further analysis or a particular solution for implementation.

3.7 Constraint Handling in EMO

The constraint handling method modifies the binary tournament selection, where two solutions are picked from the population and the better solution is chosen. In the presence of constraints, each solution can be either feasible or infeasible. Thus, there may be at most three situations: (i) both solutions are feasible, (ii) one is feasible and other is not, and (iii) both are infeasible. We consider each case by simply redefining the domination principle as follows (we call it the *constrained-domination* condition for any two solutions $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$):

Definition 3. *A solution $\mathbf{x}^{(i)}$ is said to 'constrained-dominate' a solution $\mathbf{x}^{(j)}$ (or $\mathbf{x}^{(i)} \preceq_c \mathbf{x}^{(j)}$), if any of the following conditions are true:*

1. *Solution $\mathbf{x}^{(i)}$ is feasible and solution $\mathbf{x}^{(j)}$ is not.*
2. *Solutions $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ are both infeasible, but solution $\mathbf{x}^{(i)}$ has a smaller constraint violation, which can be computed by adding the normalized violation of all constraints:*

$$\mathrm{CV}(\mathbf{x}) = \sum_{j=1}^{J} \langle \bar{g}_j(\mathbf{x}) \rangle + \sum_{k=1}^{K} \mathrm{abs}(\bar{h}_k(\mathbf{x})),$$

   where $\langle \alpha \rangle$ is $-\alpha$, if $\alpha < 0$ and is zero, otherwise. The normalization is achieved with the population minimum ($\langle g_j \rangle_{\min}$) and maximum ($\langle g_j \rangle_{\max}$) constraint violations: $\bar{g}_j(\mathbf{x}) = (\langle g_j(\mathbf{x}) \rangle - \langle g_j \rangle_{\min})/(\langle g_j \rangle_{\max} - \langle g_j \rangle_{\min})$.
3. *Solutions $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ are feasible and solution $\mathbf{x}^{(i)}$ dominates solution $\mathbf{x}^{(j)}$ in the usual sense (Definition 1).*

The above change in the definition requires a minimal change in the NSGA-II procedure described earlier. Figure 3.10 shows the non-domination fronts on a six-membered population due to the introduction of two constraints (the minimization problem is described as CONSTR elsewhere (Deb, 2001)). In the absence of the constraints, the non-domination fronts (shown by dashed lines) would have been `((1,3,5), (2,6), (4))`, but in their presence, the new fronts are `((4,5), (6), (2), (1), (3))`. The first non-domination front consists of the "best" (that is, non-dominated and feasible) points from the population and any feasible point lies on a better non-domination front than an infeasible point.

3.8 Test Problems with Known Optima

After the initial three non-elitist EMO procedures came out around 1993-94, a plethora of EMO algorithms was suggested starting around 2000. Since EMO algorithms are heuristic based, it is essential to compare them against each other and against true optimal solutions, so as to test the algorithms to investigate if they are adequate enough to handle different vagaries of problem

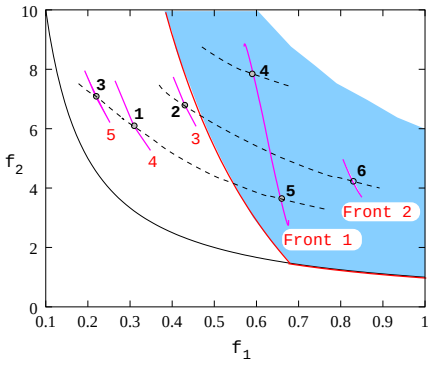


Fig. 3.10. Non-constrained-domination fronts.

difficulties real-world problems would offer. Since the purpose of EMO algorithms is to find a set of well-converged and well-spread solutions on or as close to the Pareto-optimal front as possible, it was necessary to develop numerical test problems for which the location and extent of the Pareto-optimal front are exactly known and test the performance of EMO algorithms on such test problems. The availability of test problems has not only boosted the research in EMO but has also brought researchers closer together in understanding how a population-based EMO solves multi-objective optimization problems.

Veldhuizen and Lamont (1998) collected a set of test problems from various literature including classical multi-objective optimization books. In 1999, Deb (1999b) devised a simple procedure of designing two-objective optimization problems in which different kinds of difficulties commonly found and studied in single-objective EO problems were introduced in a multi-objective optimization problem. A later study (Zitzler *et al.*, 2000) used Deb's original idea and suggested six test problems (now popularly known as Zitzler-Deb-Thiele or ZDT problems) which tested a multi-objective optimization for the following complexities:

1. Effect of increasing the number of decision variables (all six ZDT problems),
2. effect of convexity and non-convexity of the Pareto-optimal front (ZDT1 versus ZDT2),
3. effect of discontinuities and disconnectedness in the Pareto-optimal front (ZDT1 or ZDT2 versus ZDT3),
4. effect of multiple local Pareto-optimal fronts in arriving at the global Pareto-optimal front (ZDT1 versus ZDT4),
5. effect of *isolation* (presence of potentially bad solutions in the neighborhood of Pareto-optimal solutions) and *deception* (presence of a wider and

a specific type of basin of attraction for the local Pareto-optimal front) (ZDT1 versus ZDT5), and
6. effect of non-uniform density of points (on the objective space) across the Pareto-optimal front (ZDT2 versus ZDT6).

ZDT problems served the EMO research for quite many years since their suggestion in 2000. To introduce further complexities, Okabe *et al.* (2004) and Huband *et al.* (2005) have suggested test problems with strong correlations among variables. An extension of ZDT test problems to include correlations among variables was suggested in the original study Deb (1999b) and specific test problems are suggested recently (Deb *et al.*, 2006a; Igel *et al.*, 2007).

A set of nine test problems scalable to the number of objectives and variables were suggested in 2002 (Deb *et al.*, 2002c, 2005). Some of these problems, (DTLZ1 to DTLZ7 minimization problems) use a bottom-up strategy in which the functional shape of the Pareto-optimal front is first assumed and then an objective space is constructed with the desired Pareto-optimal front as the lower boundary. Thereafter, a mapping between the decision space and the objective space is constructed to introduce different kinds of complexities which can cause an EMO procedure sustained difficulties in converging and finding a widely distributed set of solutions. The remaining two test problems (DTLZ8 and DTLZ9) use a principle of starting with a hyperbox as the objective space and then introduces a number of constraints which eliminate portions of the hyperbox to help generate a Pareto-optimal front. These test problems have served EMO researchers in various ways, particularly in developing better algorithms and in testing the performance of their algorithms in solving various multi-objective optimization problems (Deb *et al.*, 2006a; Igel *et al.*, 2007).

In addition, there also exist a number of constrained test problems with non-linear objective functions and constraints. These so-called CTP problems (Deb, 2001; Deb *et al.*, 2001) introduce different complexities: disconnected feasible objective space, long narrow feasible objective space to reach to Pareto-optimal points, and others. The purpose of these test problems was to simulate different difficulties which real-world problems may have. The problems provide a tunable degree of such difficulties and this way, if an algorithm is capable of negotiating them well in test problems, the algorithm is also expected to perform well in a real-world scenario having similar difficulties.

3.9 Performance Measures and Evaluation of EMO Methodologies

When algorithms are developed and test problems with known Pareto-optimal fronts are available, an important task is to have one or more performance measures with which the EMO algorithms can be evaluated. Thus, a major

focus of EMO research has been spent in developing different performance measures and is a major area of EMO research. Chapter **??** presents a detail account of this topic. Since the focus in an EMO task is multi-faceted – convergence to the Pareto-optimal front and diversity of solutions along the entire front, it is also expected that one performance measure to evaluate EMO algorithms will be unsatisfactory. In the early years of EMO research, three different sets of performance measures were used:

1. Metrics evaluating convergence to the known Pareto-optimal front (such as error ratio, distance from reference set, etc.),
2. Metrics evaluating spread of solutions on the known Pareto-optimal front (such as spread, spacing, etc.), and
3. Metrics evaluating certain combinations of convergence and spread of solutions (such as hypervolume, coverage, R-metrics, etc.).

Some of these metrics are described in texts (Coello *et al.*, 2002; Deb, 2001). A detailed study (Knowles and Corne, 2002) comparing most existing performance metrics based on out-performance relations has concluded that R-metrics suggested by Hansen and Jaskiewicz (1998) are the best. However, a recent study has argued that a single unary performance measure (any of the first two metrics described above in the enumerated list) cannot adequately determine a true winner, as both aspects of convergence and diversity cannot be measured by a single performance metric (Zitzler *et al.*, 2003). That study also concluded that binary performance metrics (indicating usually two different values when a set of solutions A is compared against B and B is compared against A), such as epsilon-indicator, binary hypervolume indicator, utility indicators R1 to R3, etc., are better measures for multi-objective optimization. The flip side is that the binary metrics computes $M(M-1)$ performance values for two algorithms in an M-objective optimization problem, by analyzing all pair-wise performance comparisons, thereby making them difficult to use in practice. Importantly, these performance measures have allowed researchers to use them directly as fitness measures within indicator based EAs (IBEAs) (Zitzler and Künzli, 2004). In addition, unary and binary attainment indicators of Fonseca and Fleming (1996); Fonseca *et al.* (2005) are of great importance.

3.10 Other Current EMO Research and Practices

With the initial fast development of efficient EMO procedures, availability of free and commercial softwares (some of which are described in the Appendix), and applications to a wide variety of problems, EMO research and application is in its peak at the current time. It is difficult to cover every aspect of current research in a single paper. Here, we outline four main broad areas of research and application.

3.10.1 Hybrid EMO procedures

The search operators used in EMO are heuristic based. Thus these methodologies are not guaranteed to find Pareto-optimal solutions with a finite number of solution evaluations in an arbitrary problem. However, as discussed in this chapter, EMO methodologies provide adequate emphasis to currently non-dominated and isolated solutions so that population members progress towards the Pareto-optimal front iteratively. To make the overall procedure faster and to perform the task with a more theoretical basis, EMO methodologies can be combined with mathematical optimization techniques having local convergence properties. A simple-minded approach would be to start the process with an EMO and the solutions obtained from EMO can be improved by optimizing a composite objective derived from multiple objectives to ensure a good spread by using a local search technique. Another approach would be to use a local search technique as a mutation-like operator in an EMO so that all population members are at least guaranteed local optimal solutions. To save computational time, the local search based mutation can be performed after a few generations. In single-objective EA research, hybridization of EAs is common for ensuring an optima, it is time that more studies on developing hybrid EMO are pursued to ensure finding true Pareto-optimal solutions.

3.10.2 EMO and Decision-making

This book is designed to cover this topic in detail. It will suffice to point out in this chapter that finding a set of Pareto-optimal solutions by using an EMO fulfills only one aspect of multi-objective optimization, as choosing a particular solution for an implementation is the remaining decision-making task which is also equally important. In the view of the author, the decision-making task can be considered from two main aspects:

1. **Generic consideration:** There are some aspects which most practical users would like to use in narrowing down their choice. For example, in the presence of uncertainties in decision variables and/or problem parameters, the users are usually interested in finding *robust* solutions which demonstrate a insensitiveness in objective variation due to a perturbation in decision variable values or parameters. In the presence of such variations, no one is interested in Pareto-optimal but sensitive solutions. Practitioners are interested in sacrificing a global optimal solution to achieve a robust solution which when mapped to the objective space lie on a relatively flat part or which lie away from the constraint boundaries. In such scenarios and in the presence of multiple objectives, instead of finding the global Pareto-optimal front, the user may be interested in finding the *robust front* which may be partially or totally different from the Pareto-optimal front. A couple of definitions for a robust front are discussed in Chapter **??**. A recent study has developed a modified EMO procedure for finding the robust front in a problem (Deb and Gupta, 2005).

In addition, instead of finding the entire Pareto-optimal front, the users may be interested in finding some specific solutions on the Pareto-optimal front, such as *knee* points (requiring a large sacrifice in at least one objective to achieve a small gain in another thereby making it discouraging to move out from a knee point (Branke *et al.*, 2004)), Pareto-optimal points depicting certain pre-specified relationship between objectives, Pareto-optimal points having multiplicity (say, at least two or more solutions in the decision variable space mapping to identical objective values), Pareto-optimal solutions which do not lie close to variable boundaries, Pareto-optimal points having certain mathematical properties, such as all Lagrange multipliers having more or less identical magnitude – a condition often desired to have an equal importance to all constraints, and others. These considerations are motivated from the fundamental and practical aspects of optimization and may be applied to most multi-objective problem solving tasks, without any consent of a decision-maker.

2. **Subjective consideration:** In this category, any problem-specific information can be used to narrow down the choices and the process may even lead to a single preferred solution at the end. Most decision-making procedures use some preference information (utility functions, reference points (Wierzbicki, 1980), reference directions (Korhonen and Laakso, 1986), marginal rate of return and a host of other considerations (Miettinen, 1999)) to select a subset of Pareto-optimal solutions. This book is dedicated to the discussion of many such multi-criteria decision analysis (MCDA) tools and collaborative suggestions of using EMO with such MCDA tools. Some hybrid EMO and MCDA algorithms are also suggested in the recent past (Deb *et al.*, 2006b; Deb and Kumar, 2007b,a; Thiele *et al.*, 2007; Luque *et al.*, 2006).

3.10.3 Multi-objectivization

Interestingly, the act of finding multiple trade-off solutions using an EMO procedure has found its application outside the realm of solving multi-objective optimization problems per se. The concept of finding near-optimal trade-off solutions is applied to solve other kinds of optimization problems as well. For example, the EMO concept is used to solve constrained single-objective optimization problems by converting the task into a two-objective optimization task of additionally minimizing an aggregate constraint violation (Coello, 2000). This eliminates the need to specify a penalty parameter while using a penalty based constraint handling procedure. If viewed this way, the usual penalty function approach used in classical optimization studies is a special weighted-sum approach to the bi-objective optimization problem of minimizing the objective function and minimizing the constraint violation, for which the weight vector is a function of penalty parameter. A well-known difficulty in genetic programming studies, called the 'bloating', arises due to the continual increase in size of genetic programs with iteration. The reduction of

bloating by minimizing the size of programs as an additional objective helped find high-performing solutions with a smaller size of the code (Bleuler *et al.*, 2001). Minimizing the intra-cluster distance and maximizing inter-cluster distance simultaneously in a bi-objective formulation of a clustering problem is found to yield better solutions than the usual single-objective minimization of the ratio of the intra-cluster distance to the inter-cluster distance (Handl and Knowles, 2007). An EMO is used to solve minimum spanning tree problem better than a single-objective EA (Neumann and Wegener, 2005). A recent edited book (Knowles *et al.*, 2008) describes many such interesting applications in which EMO methodologies have helped solve problems which are otherwise (or traditionally) not treated as multi-objective optimization problems.

3.10.4 Applications

EMO methodologies including other a posteriori multi-objective optimization methods must be applied to more interesting real-world problems to demonstrate the utility of finding multiple trade-off solutions. Although some recent studies are finding that EMO procedures are not computationally efficient to find multiple and widely distributed sets of solutions on problems having a large number of objectives (say more than five objectives) (Deb and Saxena, 2006; Corne and Knowles, 2007), EMO procedures are still applicable in very large problems if the attention is changed to finding only a preferred region on the Pareto-optimal front, instead of the complete front. Some such preference based EMO studies (Deb *et al.*, 2006b; Deb and Kumar, 2007a; Branke and Deb, 2004) are applied to 10 or more objectives. In certain many-objective problems, the Pareto-optimal front can be low-dimensional mainly due to the presence of redundant objectives and EMO procedures can again be effective in solving such problems (Deb and Saxena, 2006; Brockhoff and Zitzler, 2007). In addition, the use of reliability based EMO (Deb *et al.*, 2007) and robust EMO (Deb and Gupta, 2005) procedures are ready to be applied to real-world multi-objective design optimization problems. Application studies are also of interest from the point of demonstrating how an EMO procedure and a subsequent MCDA approach can be combined in an iterative manner together to solve a multi-criteria decision making problem. Such efforts may lead to development of GUI-based softwares and approaches for solving the task and will demand addressing other important issues such as visualization of multi-dimensional data, parallel implementation of EMO and MCDA procedures, meta-modeling approaches, and others.

Besides solving real-world multi-objective optimization problems, EMO procedures are also found to be useful for a knowledge discovery task related to a better understanding of a problem. After a set of trade-off solutions are found by an EMO, these solutions can be compared against each other to unveil interesting principles which are common to all these solutions. These common properties among high-performing solutions will provide useful insights about what makes a solution optimal in a particular problem. Such

useful information mined from the obtained EMO trade-off solutions have been discovered in many real-world engineering design problems in the recent past and is termed as the task of 'innovization' (Deb and Srinivasan, 2006).

3.11 Conclusions

This chapter has provided a brief introduction to the fast-growing field of multi-objective optimization based on evolutionary algorithms. First, the principles of single-objective evolutionary optimization (EO) techniques have been discussed so that readers can visualize the differences between evolutionary optimization and classical optimization methods. Although the main difference seems to be in the population approach, EO methodologies do not use any derivative information and they possess a parallel search ability through their operators which makes them computationally efficient procedures.

The EMO principle of handling multi-objective optimization problems is to first attempt to find a set of Pareto-optimal solutions and then choose a preferred solution. Since an EO uses a population of solutions in each iteration, EO procedures are potentially viable techniques to capture a number of trade-off near-optimal solutions in a single simulation run. Thus, EMO procedures work in achieving two goals: (i) convergence to as close to the Pareto-optimal front as possible and (ii) maintenance of a well-distributed set of trade-off solutions. This chapter has described a number of popular EMO methodologies, presented some simulation studies on test problems, and discussed how EMO principles can be useful in solving real-world multi-objective optimization problems through a case study of spacecraft trajectory optimization.

Since early EMO research concentrated on finding a set of well-converged and well-distributed set of near-optimal trade-off solutions, EMO researchers concentrated on developing better and computationally faster algorithms by developing scalable test problems and adequate performance metrics to evaluate EMO algorithms.

Finally, this chapter has discussed the potential of EMO and its current research activities. Interestingly, in addition to EMO, these applications can also be achieved with a posteriori MCDM techniques. Besides their routine applications in solving multi-objective optimization problems, EMO and a posteriori MCDM methodologies are capable of solving other types of optimization problems, such as single-objective constrained optimization, clustering problems etc. in a better manner than they are usually solved. EMO and a posteriori MCDM methodologies are capable of unveiling important hidden knowledge about what makes a solution optimal in a problem. EMO techniques are increasingly being found to have tremendous potential to be used in conjunction with interactive multiple criterion decision making tasks in not only finding a set of optimal solutions but also to aid in selecting a preferred solution at the end.

Before closing the chapter, we provide some useful information about the EMO research and literature in the Appendix.

Acknowledgement

The author acknowledges the support of the Academy of Finland and Foundation of Helsinki School of Economics (Grant # 118319).

References

Babu, B. and Jehan, M. M. L. (2003). Differential Evolution for Multi-Objective Optimization. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC-2003)*, pages 2696–2703. Piscataway. NJ: IEEE Press.

Bäck, T., Fogel, D., and Michalewicz, Z., editors (1997). *Handbook of Evolutionary Computation*. Bristol: Institute of Physics Publishing and New York: Oxford University Press.

Bleuler, S., Brack, M., and Zitzler, E. (2001). Multiobjective genetic programming: Reducing bloat using spea2. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC-2001)*, pages 536–543. Piscataway. NJ: IEEE Press..

Branke, J. and Deb, K. (2004). Integrating user preferences into evolutionary multi-objective optimization. In *Knowledge Incorporation in Evolutionary Computation*, pages 461–477. Hiedelberg: Springer.

Branke, J., Deb, K., Dierolf, H., and Osswald, M. (2004). Finding knees in multi-objective optimization. In *Parallel Problem Solving from Nature (PPSN-VIII)*, pages 722–731. Heidelberg: Springer.

Brockhoff, D. and Zitzler, E. (2007). Dimensionality Reduction in Multiobjective Optimization: The Minimum Objective Subset Problem. In *Operations Research Proceedings 2006*, pages 423–429. Heidelberg: Springer.

Coello, C. A. C. (2000). Treating objectives as constraints for single objective optimization. *Engineering Optimization*, **32**(3), 275–308.

Coello, C. A. C. and Lechuga, M. S. (2002). MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. In *Congress on Evolutionary Computation (CEC-2002)*, pages 1051–1056. Piscataway, NJ: IEEE Press.

Coello, C. A. C. and Toscano, G. (2000). A micro-genetic algorithm for multi-objective optimization. Technical Report Lania-RI-2000-06, Laboratoria Nacional de Informatica Avanzada, Xalapa, Veracruz, Mexico.

Coello, C. A. C., VanVeldhuizen, D. A., and Lamont, G. (2002). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Boston, MA: Kluwer.

Coello, C. A. C., Aguirre, A. H., and Zitzler, E., editors (2005). In *Third International Conference on Evolutionary Multi-Criterion Optimization (EMO-2005)*. Berlin: Springer.

Corne, D. W. and Knowles, J. D. (2007). Techniques for highly multiobjective optimisation: some nondominated points are better than others. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation (GECCO-2007)*, pages 773–780. New York: The Association of Computing Machinery (ACM).

Corne, D. W., Knowles, J. D., and Oates, M. (2000). The Pareto envelope-based selection algorithm for multiobjective optimization. In *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature VI (PPSN-VI)*, pages 839–848. Heidelberg: Springer.

Corne, D. W., Jerram, N. R., Knowles, J. D., and Oates, M. J. (2001). PESA-II: Region-based selection in evolutionary multiobjective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 283–290. San Mateo, CA: Morgan Kaufmann.

Coverstone-Carroll, V., Hartmann, J. W., and Mason, W. J. (2000). Optimal multi-objective low-thurst spacecraft trajectories. *Computer Methods in Applied Mechanics and Engineering*, **186**(2–4), 387–402.

Deb, K. (1999a). An introduction to genetic algorithms. *Sādhanā*, **24**(4), 293–315.

Deb, K. (1999b). Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation Journal*, **7**(3), 205–230.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley.

Deb, K. and Agrawal, S. (1999). A niched-penalty approach for constraint handling in genetic algorithms. In *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA-99)*, pages 235–243. Heidelberg: Springer.

Deb, K. and Gupta, H. (2005). Searching for robust Pareto-optimal solutions in multi-objective optimization. In *Proceedings of the Third Evolutionary Multi-Criteria Optimization Conference (EMO-2005)*, pages 150–164, Heidelberg: Springer.

Deb, K. and Kumar, A. (2007a). Interactive evolutionary multi-objective optimization and decision-making using reference direction method. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2007)*, pages 781–788. New York: The Association of Computing Machinery (ACM).

Deb, K. and Kumar, A. (2007b). Light beam search based multi-objective optimization using evolutionary algorithms. In *Proceedings of the Congress on Evolutionary Computation (CEC-2007)*, pages 2125–2132. Piscataway, NJ: IEEE Press.

Deb, K. and Saxena, D. (2006). Searching for pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems. In *Proceedings of the World Congress on Computational Intelligence (WCCI-2006)*, pages 3352–3360. Piscataway, NJ: IEEE Press.

Deb, K. and Srinivasan, A. (2006). Innovization: Innovating design principles through optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2006)*, pages 1629–1636. New York: The Association of Computing Machinery (ACM).

Deb, K. and Tiwari, S. (2008). Omni-optimizer: A generic evolutionary algorithm for global optimization. *European Journal of Operations Research (EJOR)*. **185**(3), 1062–1087.

Deb, K., Pratap, A., and Meyarivan, T. (2001). Constrained test problems for multi-objective evolutionary optimization. In *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO-2001)*, pages 284–298. Heidelberg: Springer.

Deb, K., Anand, A., and Joshi, D. (2002a). A computationally efficient evolutionary algorithm for real-parameter optimization. *Evolutionary Computation Journal*, **10**(4), 371–395.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2002b). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, **6**(2), 182–197.

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2002c). Scalable multi-objective optimization test problems. In *Proceedings of the Congress on Evolutionary Computation (CEC-2002)*, pages 825–830. Piscataway, NJ: IEEE Press.

Deb, K., Reddy, A. R., and Singh, G. (2003a). Optimal scheduling of casting sequence using genetic algorithms. *Journal of Materials and Manufacturing Processes*, **18**(3), 409–432.

Deb, K., Mohan, M., and Mishra, S. (2003b). Towards a quick computation of well-spread pareto-optimal solutions. In *Proceedings of the Second Evolutionary Multi-Criterion Optimization Conference (EMO-2003)*, pages 222–236. Heidelberg: Springer.

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2005). Scalable test problems for evolutionary multi-objective optimization. In A. Abraham, L. Jain, and R. Goldberg, editors, *Evolutionary Multiobjective Optimization*, pages 105–145. London: Springer-Verlag.

Deb, K., Sinha, A., and Kukkonen, S. (2006a). Multi-objective test problems, linkages and evolutionary methodologies. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2006)*, pages 1141–1148. New York: The Association of Computing Machinery (ACM).

Deb, K., Sundar, J., Uday, N., and Chaudhuri, S. (2006b). Reference point based multi-objective optimization using evolutionary algorithms. *International Journal of Computational Intelligence Research (IJCIR)*, **2**(6), 273–286.

Deb, K., Padmanabhan, D., Gupta, S., and Mall, A. K. (2007). Reliability-based multi-objective optimization using evolutionary algorithms. In *Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization (EMO-2007)*, pages 66–80. Heidelberg: Springer.

Deb, K., Tiwari, R., Dixit, M., and Dutta, J. (2007). Finding trade-off solutions close to kkt points using evolutionary multi-objective optimization. In *Proceedings of the Congress on Evolutionary Computation (CEC-2007)*. pages 2109–2116. Piscataway, NJ: IEEE Press.

Fogel, L. J., Owens, A. J., and Walsh, M. J. (1966). *Artificial Intelligence Through Simulated Evolution*. New York: Wiley.

Fonseca, C., Fleming, P., Zitzler, E., Deb, K., and Thiele, L. (2003). *Proceedings of the Second Evolutionary Multi-Criterion Optimization Conference (EMO-2003)*. Heidelberg: Springer.

Fonseca, C. M. and Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion, and generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423. San Mateo, CA: Morgan Kaufmann.

Fonseca, C. M. and Fleming, P. J. (1996). On the performance assessment and comparison of stochastic multiobjective optimizers. In *Parallel Problem Solving from Nature (PPSN IV)*, pages 584–593. Berlin: Springer.

Fonseca, C. M., da Fonseca, V. G., and Paquete, L. (2005). Exploring the performance of stochastic multiobjective optimisers with the second-order attainment function. In *Third International Conference on Evolutionary Multi-Criterion Optimization, (EMO-2005)*, pages 250–264. Heidelberg: Springer.

Gen, M. and Cheng, R. (1997). *Genetic Algorithms and Engineering Design*. New York: Wiley.

Goldberg, D. E. (1989). *Genetic Algorithms for Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley.

Goldberg, D. E. and Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, pages 41–49. New Jersy: Lawrence Erlbaum.

Goldberg, D. E., Deb, K., and Thierens, D. (1993). Toward a better understanding of mixing in genetic algorithms. *Journal of the Society of Instruments and Control Engineers (SICE)*, **32**(1), 10–16.

Gravel, M., Price, W. L., and Gagné, C. (2002). Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic. *European Journal of Operational Research*, **143**(1), 218–229.

Handl, J. and Knowles, J. D. (2007). An evolutionary approach to multiobjective clustering. *IEEE Transactions on Evolutionary Computation*, **11**(1), 56–76.

Hansen, M. P. and Jaskiewicz, A. (1998). Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Lyngby: Institute of Mathematical Modelling, Technical University of Denmark.

Herrera, F., Lozano, M., and Verdegay, J. L. (1998). Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. *Artificial Intelligence Review*, **12**(4), 265–319.

Holland, J. H. (1962). Concerning efficient adaptive systems. In *Self-Organizing Systems*, pages 215–230. Washington: Spartan Press.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor: MIT Press.

Horn, J., Nafploitis, N., and Goldberg, D. (1994). A niched Pareto genetic algorithm for multi-objective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, pages 82–87. Piscataway, NJ: IEEE Press.

Huband, S., Barone, L., While, L., and Hingston, P. (2005). A scalable multi-objective test problem toolkit. In *Proceedings of the Evolutionary Multi-Criterion Optimization (EMO-2005)*. Heidelberg: Springer.

Igel, C., Hansen, N., and Roth, S. (2007). Covariance matrix adaptation for multi-objective optimization evolutionary computation. *Evolutionary Computation Journal*, **15**(1), 1–28.

Jansen, T. and Wegener, I. (2001). On the utility of populations. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001)*, pages 375–382. San Mateo, CA: Morgan Kaufmann.

Kennedy, J., Eberhart, R. C., and Shi, Y. (2001). *Swarm intelligence*. San Mateo, CA: Morgan Kaufmann.

Knowles, J. D. and Corne, D. W. (2000). Approximating the non-dominated front using the Pareto archived evolution strategy. *Evolutionary Computation Journal*, **8**(2), 149–172.

Knowles, J. D. and Corne, D. W. (2002). On metrics for comparing nondominated sets. In *Congress on Evolutionary Computation (CEC-2002)*, pages 711–716. Piscataway, NJ: IEEE Press.

Knowles, J. D., Corne, D. W., and Deb, K. (2008). *Multiobjective problem solving from nature*. Heidelberg: Springer.

Korhonen, P. and Laakso, J. (1986). A visual interactive method for solving the multiple criteria problem. *European Journal of Operational Reseaech*, **24**, 277–287.

Koza, J. R. (1992). *Genetic Programming : On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press.

Kung, H. T., Luccio, F., and Preparata, F. P. (1975). On finding the maxima of a set of vectors. *Journal of the Association for Computing Machinery*, **22**(4), 469–476.

Laumanns, M., Thiele, L., Deb, K., and Zitzler, E. (2002). Combining convergence and diversity in evolutionary multi-objective optimization. *Evolutionary Computation*, **10**(3), 263–282.

Loughlin, D. H. and Ranjithan, S. (1997). The neighborhood constraint method: A multiobjective optimization technique. In *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 666–673. New York: The Association of Computing Machinery (ACM).

Luque, M., Miettinen, K., Eskelinen, P., and Ruiz, F. (2006). Three different ways for incorporating preference information in interactive reference point based methods. Technical Report W-410, Helsinki School of Economics, Helsinki, Finland.

McMullen, P. R. (2001). An ant colony optimization approach to addessing a JIT sequencing problem with multiple objectives. *Artificial Intelligence in Engineering*, **15**, 309–317.

Michalewicz, Z. (1992). *Genetic Algorithms + Data Structures = Evolution Programs*. Berlin: Springer-Verlag.

Miettinen, K. (1999). *Nonlinear Multiobjective Optimization*. Boston: Kluwer.

Mostaghim, S. and Teich, J. (2003). Strategies for Finding Good Local Guides in Multi-objective Particle Swarm Optimization (MOPSO). In *2003 IEEE Swarm Intelligence Symposium Proceedings*, pages 26–33, Piscataway, NJ: IEEE Press.

Neumann, F. and Wegener, I. (2005). Minimum spanning trees made easier via multi-objective optimization. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 763–769, New York: The Association of Computing Machinery (ACM).

Obayashi, S., Deb, K., Poloni, C., Hiroyasu, T., and Murata, T., editors (2007). *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007, Matsushima, Japan, March 5-8, 2007, Proceedings*, Heidelberg: Springer.

Okabe, T., Jin, Y., Olhofer, M., and Sendhoff, B. (2004). On test functions for evolutionary multi-objective optimization. In *Parallel Problem Solving from Nature (PPSN VIII)*, pages 792–802. Heidelberg: Springer.

Osyczka, A. (2002). *Evolutionary algorithms for single and multicriteria design optimization*. Heidelberg: Physica-Verlag.

Price, K. V., Storn, R., and Lampinen, J. (2005). *Differential Evolution: A Practical Approach to Global Optimization*. Berlin: Springer-Verlag.

Radcliffe, N. J. (1991). Forma analysis and random respectful recombination. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 222–229.

Rechenberg, I. (1965). Cybernetic solution path of an experimental problem. Royal Aircraft Establishment, Library Translation Number 1122, Farnborough, UK.

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*. Stuttgart: Frommann-Holzboog Verlag.

Rosenberg, R. S. (1967). *Simulation of Genetic Populations with Biochemical Properties*. Ph.D. thesis, Ann Arbor, MI, University of Michigan.

Rudolph, G. (1994). Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Network*, **5**(1), 96–101.

Sasaki, D., Morikawa, M., Obayashi, S., and Nakahashi, K. (2001). Aerodynamic shape optimization of supersonic wings by adaptive range multiobjective genetic algorithms. In *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001)*, pages 639–652. Heidelberg: Springer.

Sauer, C. G. (1973). Optimization of multiple target electric propulsion trajectories. In *AIAA 11th Aerospace Science Meeting*. Paper Number 73-205.

Schaffer, J. D. (1984). *Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms*. Ph.D. thesis, Nashville, TN: Vanderbilt University.

Schwefel, H.-P. (1968). Projekt MHD-Staustrahlrohr: Experimentelle optimierung einer zweiphasendüse, teil I. Technical Report 11.034/68, 35, AEG Forschungsinstitut, Berlin.

Schwefel, H.-P. (1995). *Evolution and Optimum Seeking*. New York: Wiley.

Srinivas, N. and Deb, K. (1994). Multi-objective function optimization using non-dominated sorting genetic algorithms. *Evolutionary Computation Journal*, **2**(3), 221–248.

Storn, R. and Price, K. (1997). Differential evolution – A fast and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, **11**, 341–359.

Thiele, L., Miettinen, K., Korhonen, P., and Molina, J. (2007). A preference-based interactive evolutionary algorithm for multiobjective optimization. Technical Report Working Paper Number W-412, Helsinki School of Economics, Finland.

Veldhuizen, D. V. and Lamont, G. B. (1998). Multiobjective evolutionary algorithm research: A history and analysis. Technical Report TR-98-03, Dayton, OH: Department of Electrical and Computer Engineering, Air Force Institute of Technology.

Veldhuizen, D. V. and Lamont, G. B. (2000). Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary Computation Journal*, **8**(2), 125–148.

Vose, M. D., Wright, A. H., and Rowe, J. E. (2003). Implicit parallelism. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2003)*, pages 1505–1517. Heidelberg: Springer.

Wierzbicki, A. P. (1980). The use of reference objectives in multiobjective optimization. In G. Fandel and T. Gal, editors, *Multiple Criteria Decision Making Theory and Applications*, pages 468–486. Berlin: Springer-Verlag.

Zitzler, E. and Künzli, S. (2004). Indicator-Based Selection in Multiobjective Search. In *Conference on Parallel Problem Solving from Nature (PPSN VIII)*, pages 832–842. Heidelberg: Springer.

Zitzler, E. and Thiele, L. (1998). Multiobjective optimization using evolutionary algorithms – A comparative case study. In *Parallel Problem Solving from Nature V (PPSN-V)*, pages 292–301. Heidelberg: Springer.

Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, **3**(4), 257–271.

Zitzler, E., Deb, K., and Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation Journal*, **8**(2), 125–148.

Zitzler, E., Deb, K., Thiele, L., Coello, C. A. C., and Corne, D. W. (2001a). *Proceedings of the First Evolutionary Multi-Criterion Optimization Conference (EMO-2001)*. Heidelberg: Springer.

Zitzler, E., Laumanns, M., and Thiele, L. (2001b). SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece. International Center for Numerical Methods in Engineering (CMINE).

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., and Fonseca, V. G. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, **7**(2), 117–132.

Appendex: EMO Repository

Here, we outline some dedicated literature in the area of multi-objective optimization. Further details can be found from `http://www.lania.mx/~ccoello/EMOO/`.

Some Relevant Books in Print

- A. Abraham, L. C. Jain and R. Goldberg. *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, London: Springer-Verlag, 2005.
  This is a collection of the latest state-of-the-art theoretical research, design challenges and applications in the field of EMO.
- C. A. C. Coello, D. A. VanVeldhuizen, and G. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Boston, MA: Kluwer Academic Publishers, 2002.
  A good reference book with a good citation of most EMO studies. A revised version is in print.
- Y. Collette and P. Siarry. *Multiobjective Optimization: Principles and Case Studies*, Berlin: Springer, 2004.
  This book describes multi-objective optimization methods including EMO and decision-making, and a number of engineering case studies.
- K. Deb. *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001. (Third edition, with exercise problems)
  A comprehensive text-book introducing the EMO field and describing major EMO methodologies and salient research directions.
- N. Nedjah and L. de Macedo Mourelle (Eds.). *Real-World Multi-Objective System Engineering*, New York: Nova Science Publishers, 2005.
  This edited book discusses recent developments and application of multi-objective optimization including EMO.
- A. Osyczka. *Evolutionary algorithms for single and multicriteria design optimisation*, Heidelberg: Physica-Verlag, 2002.
  A book describing single and multi-objective EAs with many engineering applications.
- M. Sakawa. *Genetic Algorithms and Fuzzy Multiobjective Optimization*, Norwell, MA: Kluwer, 2002.
  This book discusses EMO for 0-1 programming, integer programming, nonconvex programming, and job-shop scheduling problems under multiobjectiveness and fuzziness.

- K. C. Tan and E. F. Khor and T. H. Lee. *Multiobjective Evolutionary Algorithms and Applications*, London, UK: Springer-Verlag, 2005.
  A book on various methods of preference-based EMO and application case studies covering areas such as control and scheduling.

Some Review Papers

- C. A. C. Coello. Evolutionary Multi-Objective Optimization: A Critical Review. In R. Sarker, M. Mohammadian and X. Yao (Eds), *Evolutionary Optimization*, pp. 117–146, Kluwer Academic Publishers, New York, 2002.
- C. Dimopoulos. A Review of Evolutionary Multiobjective Optimization Applications in the Area of Production Research. *2004 Congress on Evolutionary Computation (CEC'2004)*, IEEE Service Center, Vol. 2, pp. 1487–1494, 2004.
- S. Huband, P. Hingston, L. Barone and L. While. A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit, *IEEE Transactions on Evolutionary Computation*, Vol. 10, No. 5, pp. 477–506, 2006.

Dedicated Conference Proceedings

- S. Obayashi, K. Deb, C. Poloni, and T. Hiroyasu (Eds.)., *Evolutionary Multi-Criterion Optimization (EMO-07) Conference Proceedings*, Also available as LNCS 4403. Berlin, Germany: Springer, 2007.
- C. A. Coello Coello and A. H. Aguirre and E. Zitzler (Eds.), *Evolutionary Multi-Criterion Optimization (EMO-05) Conference Proceedings*, Also available as LNCS 3410. Berlin, Germany: Springer, 2005.
- Fonseca, C., Fleming, F., Zitzler, E., Deb, K., and Thiele, L. (Eds.), *Evolutionary Multi-Criterion Optimization (EMO-03) Conference Proceedings*. Also available as LNCS 2632. Heidelberg: Springer, 2003.
- Zitzler, E., Deb, K., Thiele, L., Coello, C. A. C. and Corne, D. (Eds.), *Evolutionary Multi-Criterion Optimization (EMO-01) Conference Proceedings*, Also available as LNCS 1993. Heidelberg: Springer, 2001.

Some Public-domain source codes

- NIMBUS: `http://www.mit.jyu.fi/MCDM/soft.html`
- NSGA-II in C: `http://www.iitk.ac.in/kangal/soft.htm`
- PISA: `http://www.tik.ee.ethz.ch/sop/pisa/`
- Shark in C++: `http://shark-project.sourceforge.net`
- SPEA2 in C++: `http://www.tik.ee.ethz.ch/~zitzler`

Some commercial codes implementing EMO

- GEATbx in Matlab (`http://www.geatbx.com/`)
- iSIGHT and FIPER from Engineous (`http://www.engineous.com/`)
- MAX from CENAERO (`http://www.cenaero.be/`)
- modeFRONTIER from Esteco (`http://www.esteco.com/`)

Index

ant-based EMO 20
asymptotic convergence 5
attainment indicators 26

bloating 28

clustering problem 28
dovar textbf 22
convex programming 3
crossover operator 4
crowding distance 17
customized algorithm 4

Differential evolution 6
differential evolution 20
DTLZ test problems 24

Elitism 7
EMO 1
  Application 20
  constraint handling 22
  decision-making 27
  elitist approach 16
  hybrid approaches 26
  non-elitist approach 14, 23
  performance metrics 25
  reliability based 29
  Repository 36
  robust Pareto-optimal frontier 27
  test problems 23
  dovar constrained test suite 25
  dovar DTLZ test suite 24
  dovar ZDT test suite 24
epsilon-indicator 26

Evolution strategy 7
Evolutionary optimization 7
evolutionary optimization 2
Evolutionary programming 7

feasible decision variable space 9
fitness function 4

Genetic algorithm 7
Genetic programming 8

hypervolume indicator 26

innovization 29

Karush-Kuhn-Tucker conditions 5
knee point 27
KUR 18

Mating Pool 8
MCDM 1
micro-GA 20
MOGA 15
MOMGA 20
multi-objective optimization
  problem 9
mutation operator 5

niching 15
NPGA 15
NSGA 16, 20
NSGA-II 16

objective space 9

PAES 16, 19
particle swarm EMO 20

performance metrics 25
PESA 19
PESA2 20
posteriori approach 13

reference direction 27
reference point 27

SBX 5
Selection 9
simulated binary crossover 5
SPEA 16, 18
SPEA2 19

termination criteria 5
test problems
  KUR 18
tournament selection 4, 22

utility indicators 26

variation operator 4
VEGA 14

ZDT problems 24
ZDT2 18