Memetic Algorithms for Multiobjective Optimization: Issues, Methods and Prospects

Joshua Knowles[1] and David Corne[2]

[1]Dept of Chemistry, UMIST, PO Box 88, Sackville St, Manchester M60 1QD
[2]Dept of Computer Science, Harrison Building, University of Exeter EX4 4QF

Summary. The concept of optimization—finding the extrema of a function that maps candidate 'solutions' to scalar values of 'quality'—is an extremely general and useful idea that can be, and is, applied to innumerable problems in science, industry, and commerce. However, the vast majority of 'real' optimization problems, whatever their origins, comprise more than one objective; that is to say, 'quality' is actually a vector, which may be composed of such distinct attributes as cost, performance, profit, environmental impact, and so forth, which are often in mutual conflict. Until relatively recently this uncomfortable truth has been (wilfully?) overlooked in the sciences dealing with optimization, but now, increasingly, the idea of *multiobjective optimization* is taking over the centre ground. Multiobjective optimization takes seriously the fact that in most problems the different components that describe the quality of a candidate solution cannot be lumped together into one representative, overall measure, at least not easily, and not before some understanding of the possible 'tradeoffs' available has been established. Hence a multiobjective optimization *algorithm* is one which deals directly with a vector objective function and seeks to find multiple solutions offering different, optimal tradeoffs of the multiple objectives. This approach raises several unique issues in optimization algorithm design, which we consider in this article, with a particular focus on memetic algorithms (MAs). We summarize much of the relevant literature, attempting to be inclusive of relatively unexplored ideas, highlight the most important considerations for the design of multiobjective MAs, and finally outline our visions for future research in this exciting area.

1 Introduction

Many important problems arising in science, industry and commerce fall very neatly into the ready-made category of *optimization problems*; that is to say, these problems are solved if we can simply find a 'solution' that maximizes or minimizes some important and measurable property or attribute, such as cost or profit. For example, we might want to find the set of routes that minimizes the distance travelled by a fleet of delivery lorries; or to find the tertiary structure of a trans-membrane protein that minimizes its free energy;

or to find the portfolio of stocks that maximizes the expected profit over the forthcoming year. Of course, solving these problems exactly might be very difficult or impossible in practice, but by applying one of numerous optimization algorithms—memetic algorithms (MAs) being one very flexible and successful possibility—very good solutions can often be found.

However, there is a caveat: maximizing or minimizing a single, lone attribute can, in many cases, be a very bad thing to do. Consider designing a car with the single objective of minimizing its weight: other desirable attributes like safety, comfort, and capacity would be severely compromised as a result. And so it is in many other generic problems: maximizing profit often leads to compromises in environmental impact or customer satisfaction; minimizing production costs often leads to decreased reliability; and minimizing planned time to completion of a project often leads to soaring costs for overrunning. Thus, it is easy to appreciate that most 'real' optimization problems involve optimizing, simultaneously, more than one single attribute.

Now, given that most problems are as we've described—'multiobjective' in nature—, what are the options for tackling them? There are basically three: (1) ignore some of the attributes entirely and just optimize one that looks most important; (2) lump all the objectives together by just adding them up or multiplying them together and then optimize the resulting function; or (3) apply a multiobjective algorithm that seeks to find *all* the solutions that are *nondominated* (we define this later but, roughly speaking, nondominated solutions are those that are optimal under *some/any* reasonable way of combining the different objective functions into a single one). The first and second options are very common and the third less so. However, (3) is rapidly gaining popularity as it is more and more understood that (1) and (2) can be very damaging in practice—solving the problem might be very satisfying to the algorithm or MA practitioner, but the resulting solution may be far from optimal when it is applied back in the real world. Thus, in this chapter, we will argue that option (3)—seeking *multiple, distinct solutions* to a problem, conferring different tradeoffs of the objectives,—is the essence of true multiobjective optimization (MOO).

Doing true multiobjective optimization with memetic algorithms requires a few salient adaptations to the normal design principles. Clearly, since we need to find multiple, distinct solutions, the design of multiobjective MAs will be heavily affected by the need to encourage and preserve *diversity*. Indeed, much of the research in evolutionary algorithms (EAs) for MOO has concerned itself primarily with this issue, but with MAs the use of local search introduces further complications for achieving diversity that must be resolved.

The goal of finding multiple solutions also dictates that the MA incorporate some means of storing the best solutions discovered. While MAs are already endowed with a population, some research in EAs for MOO has found that methods that exploit secondary populations, or *archives*, seem to perform better than single-population approaches, and *elitism* based on archives appears to be particularly effective in improving search capability. Thus, ques-

tions about how to control and use multiple populations (or non-fixed size populations) are somewhat more relevant and pressing in MOO than they are in 'normal' optimization.

A second key distinction of MOO, closely related to the need for multiple solutions, is the inherent *partial* ordering of solutions in terms of their overall quality, which characterises MOO. This impacts on many aspects of search and how it should be conducted. In particular, the simple comparison of two solutions is fraught with difficulties. Local search, which relies upon such comparisons being made, must be re-defined in some way, and there are several competing possibilities.

There are also innumerable possibilities concerning the overall organization of the search—how the set of tradeoff solutions (the nondominated set) is to be built up, over time. Very coarsely, should we try to sweep across the objective space from one 'edge' to the other, i.e. improving one combination of objectives at a time, or should we more try to push the entire 'surface' down in parallel, improving the whole currently nondominated set at once? In either case, what is the best way to exploit the population(s) and the different local searchers at our disposal?

In the remainder of this article, we will try to fill the reader in on the core issues we have but sketched here, mapping out the little that is known and has been tried so far, and speculating about where further research may be most fruitful. In section 2, some MOO applications are outlined to give some idea of their range and differing characteristics. The mathematical definition of the MOO problem is then given, and Pareto optimization is described. Section 3 visits a large number of metaheuristics for MOO and identifies concepts and strategies that, we suggest, may be useful as components in a memetic algorithm. In section 4, we elaborate on other issues in MOO research that may impact on the design and application of multiobjective MAs, including how to measure performance, how multiple populations can be used, and available test functions. Section 5 provides a focused review of existing MAs for MOO, while section 6 proposes some principles for designing more advanced MAs. The last section considers future research directions and gives some recommendations for immediate investigation.

2 A Brief Introduction to MOO

2.1 MAs and MOO: a good combination

The impressive record of memetic algorithms in producing high quality solutions in combinatorial optimization and in real-world applications (e.g. see page 220 [18]) is sometimes cited as a testament to their inherent effectiveness or robustness as black-box searchers. However, since the advent of the No Free Lunch theorems [109, 19, 21], we know that MAs, like any other search algorithm, are only really good to the extent to which they can be 'aligned'

to the specific features of an optimization problem. Nonetheless, MAs, like their forebears, EAs, do have one unassailable advantage over other more traditional search techniques: that is their *flexibility*. EAs and MAs impose no requirement for a problem to be formulated in a particular constraint language, and do not ask for the function to be differentiable, continuous, linear, separable, or of any particular data-type. Rather, they can be applied to *any* problem for which the following are available: (1) some (almost) any way to encode a candidate solution to the problem, and (2) some means of computing the quality of any such encoded solution—the so-called objective function.

This flexibility has important advantages. As has been observed in [83], there are basically two ways to solve optimization problems: one is to choose some traditional technique and then simplify or otherwise alter the problem formulation to allow the problem to be tackled using the chosen technique; the other is to leave the problem formulation in its original form and use an EA, MA, or other metaheuristic. Clearly, the latter is preferable because the answer one arrives at is (at least) to the right question, not to a question which may have been distorted (perhaps so much so as to be irrelevant to the real question), simply to fit in with the requirements of a chosen search method.

In [19], the advantages of 'leaving the problem alone' (and applying a flexible search technique) was reiterated and used to make a further, compelling point. How often are optimization problems in the real world (or from real-world origins) squeezed and stretched into the strait-jacket of a single-objective formulation, when their natural formulation is to have *multiple* objectives? Doesn't the same observation of [83] apply in this case, too? What is the effect of throwing away objectives or of combining them together as a weighted, linear sum, as is so often done? If we are to believe the EA/MA mantra about tackling problems in their original formulation, shouldn't we be tackling *multiobjective* problems in the same way?

Of course, the answer is that we should. And there are two reasons: (1) simplifying a problem *does* change it irrevocably and make it irrelevant in many cases, and (2) with EAs, including MAs, we have the capability to tackle multiobjective problems in their native form and indeed the cost of doing so is demonstrably not high.

2.2 Some example MOO problems

One could argue that engineering is the art of finding the good compromise; and indeed many problems encountered in engineering do have multiple and distinct objectives. Fortunately, we are now gradually seeing that the optimization problems being formulated in various engineering sub-disciplines are respecting the multiobjective nature of the underlying problem. For example, civil engineering tasks such as designing water networks are being seen as multiobjective optimization problems [48, 13, 14, 15], as is power

distribution [3, 4, 6, 79], and various telecommunications network optimization tasks [73, 72]. And, at the other end of the engineering spectrum, the design of various types of controllers has been aided by such an approach [2, 8, 104, 24, 38, 41] for some years now.

Scheduling and *timetabling* are two huge classes of *planning problem* that can involve a multitude of different objectives. In scheduling, problems tackled in the academic literature often consider only one objective: minimizing the makespan—the total time needed to complete all jobs. However, the reality of scheduling in factories, space programmes, engineering projects and so forth is far more complex. Reducing the makespan is undoubtedly one objective but other important ones are mean and total tardiness, mean flow time, mean waiting time, and the mean and total completion time. In addition to these objectives there are often a number of constraints. If all these constraints are modelled as 'hard', the resulting schedules can be brittle and sub-optimal. By softening some of these constraints (those which are not really inviolable) and treating them as further objectives, great gains can sometimes be made for minute sacrifices elsewhere. Frequently, the robustness of a schedule to unforeseen changes, such as late arrival times of materials, machine failures and so forth, should also be modelled. Making robustness an objective enables planners to consider fully the tradeoffs between allowing some slack, versus 'risking it' and going for the absolutely optimal schedule.

Much the same can be said for timetabling, particularly with regard to constraints. More often than not, timetabling problems are tackled as constraint satisfaction problems in which hard constraints must be satisfied and soft constraint violations should be minimized. However, the latter are usually just added together, leading to absurd situations, where, for example, the optimization algorithm 'chooses' that nineteen students having consecutive exams is better than 14 having to get up early one morning, together with 6 invigilators working through their lunch break! Fortunately, the recognition that these problems are multiobjective, and need to be tackled as such, is leading to more research in this vein: e.g. [46, 51, 59, 62] in scheduling, and [91, 12] in timetabling.

There are a whole host of other varied MOO applications emerging on a more and more frequent basis: from the training of neural networks [1, 11, 111, 93], to various design applications [92, 95, 5], to dealing with the challenges of dynamic optimization [110, 35]. The short survey presented here scratches but the surface, and the reader is directed to [32] and [16] for more comprehensive reviews.

Basic MOO definitions

An unconstrained multiobjective optimization problem can be formulated as

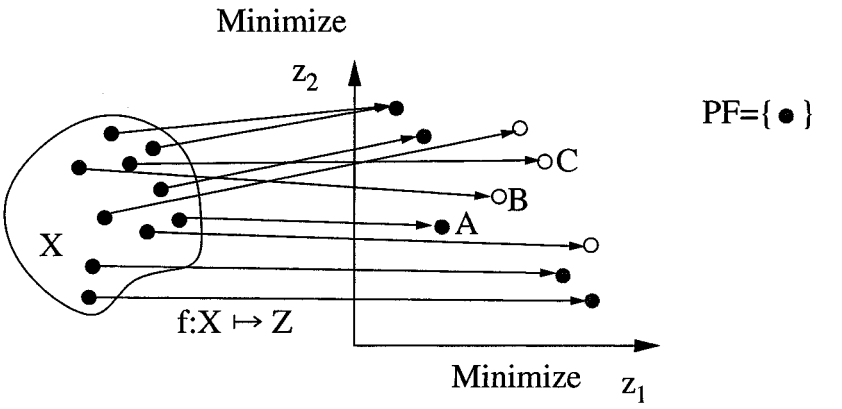

Fig. 1. An illustration of a multiobjective optimization problem with a search space X, a vector fitness function $\mathbf{f}$ that maps solutions in X to objective vectors made up of two component 'costs' z_1 and z_2 to be minimized. The solid objective vectors are nondominated and comprise the Pareto front. The solutions corresponding to these points are Pareto optimal. The relation between the three objective vectors A, B, and C is $A < B < C$

$$\begin{aligned} &\text{"minimize"} \quad \mathbf{z} = \mathbf{f}(\mathbf{x}) \\ &\text{where} \qquad \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \\ &\text{subject to} \quad \mathbf{x} \in X \end{aligned} \tag{1}$$

involving $k \geq 2$ different *objective functions* $f_i : \Re^n \mapsto \Re$ to be minimized simultaneously. Note that if f_i is to be maximized, it is equivalent to minimize $-f_i$.

The term "minimize" appears in quotes in (1) to emphasise that the exact meaning of the vector minimization must be specified before optimization can be performed. That is, we need to specify a binary relation on objective vectors in order to form a (partial) ordering of them. Although different possibilities exist, in this chapter we will be concerned only with the component-wise order relation, which forms the basis for Pareto optimization as defined below (also see figure 1).

Definition 1 *The component-wise order relation* $<$ *is defined as* $\mathbf{z}^r < \mathbf{z}^s \Leftrightarrow z_i^r \leq z_i^s, i = 1..k \wedge \mathbf{z}^r \neq \mathbf{z}^s$.

Definition 2 *A solution* $\mathbf{x}^* \in X$ *is called* **Pareto optimal** *if there is no* $\mathbf{x} \in X$ *such that* $\mathbf{f}(\mathbf{x}) < \mathbf{f}(\mathbf{x}^*)$. *If* $\mathbf{x}^*$ *is Pareto optimal,* $\mathbf{z}^* = \mathbf{f}(\mathbf{x}^*)$ *is called (globally)* **nondominated**. *The set of all Pareto optima is called the* **Pareto optimal set**, *and the set of all nondominated objective vectors is called the* **Pareto front (PF)**. *Finding an approximation to either the Pareto optimal set or the Pareto front is called* **Pareto optimization**.

More generally, Miettinen [84] defines solving a multiobjective problem as finding a Pareto optimal solution to (1) that also satisfies a *decision maker* (DM), who knows or understands something more about the problem. Such a definition brings into play the science of multi-criteria decision making (MCDM), where methods are used to model the preferences of decision makers in order to aid them in comparing and choosing solutions. Thus, according to this definition, solving a multiobjective problem, involves both search and *decision making*, and to accomplish this, one of three general approaches is normally taken:

1. *A priori* optimization
2. *A posteriori* optimization
3. Interactive optimization

In *a priori* optimization, the decision maker is consulted before search and a mathematical model of her preferences is constructed (following one of several regimes for this), and used in the search to evaluate all solutions. The best solution found, according to the model, is returned and represents the outcome of the optimization process with no further input from the DM. The drawback with such methods is obvious: decision makers find it very hard to give adequate models determining which solutions they prefer, without knowing or having any idea what it is possible to attain, and how much one objective may have to be sacrificed with respect to others. Furthermore, notice that this method, in a sense, places all the additional work associated with MOO, firmly with the DM, and leaves the search problem as seen by a search algorithm, in much the same form as for normal optimization, i.e. one solution must be found and all solutions are comparable (using the DM's *a priori* preference model). For this reason, we do not consider *a priori* optimization any further in this article, as standard MAs could be used (or trivially adapted) to this case.

A posteriori optimization approaches the multiobjective problem from the reverse angle. First, search is conducted to find the Pareto optimal set (or an approximation/representation thereof) and the DM will then choose between these alternatives by inspection (with or without using some mathematical decision-making aid). The disadvantage (according to [84]) of this approach is the difficulty DMs may have in visualizing the different alternatives and choosing between them, particularly if a large number have been generated. Nonetheless, the problem of decision-making is in our opinion definitely aided by knowing something about what solutions are possible. Thus, *a posteriori* methods move at least some of the work from the DM to the search algorithm, which now is given the task of searching for multiple different solutions. Exactly what solutions the search algorithm finds will depend upon how, internally, it evaluates solutions, but it should be oriented towards finding Pareto optima. And in order to give the DM what she needs—real *alternatives*—the Pareto optima should not be all in one region of the objective space, but should be spread as far and wide as possible. (Being more precise than this is

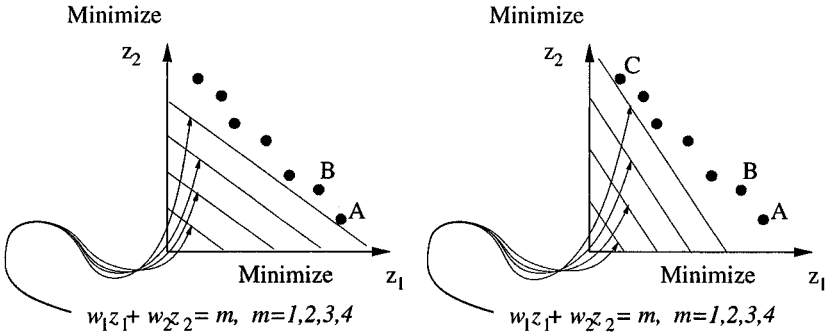

Fig. 2. Illustration of the drawbacks of scalarizing objectives using the weighted sum approach. The figures show a Pareto front and lines of equal cost under a weighted sum. In the left figure, A is the optimal solution. A slight change to the weights, slightly altering the angle of the isocost lines, as shown in the figure on the right, makes C the optimal solution. The nondominated solution B is 'non-supported' – not on the convex hull of the Pareto front. Therefore it is not optimal under any linear combination of the objectives

problematic as seen in section 4.1 where we will discuss how to evaluate different approximations to Pareto fronts). In any case, *a posteriori* optimization is the method we advocate in this article, in preference to *a priori* methods, and we assume in the remainder of the article that finding a 'good' approximation to the whole Pareto front is the goal of multiobjective optimization, leaving decision-making as a separate issue.

The interactive methods of search combine *a priori* and *a posteriori* methods in an iterative funnelling of goals, preferences and solutions discovered. These methods are probably preferable to *a posteriori* methods, since they limit the choices shown to a DM at any instant, and focus the search on a smaller area. However, we do not make more than a passing reference to them in what follows, for two reasons. First, because, so far, relatively little research in the EA community has been directed to this general approach, so it is difficult to make judgments or recommendations. And more importantly, because effectively, from a search point of view, the problem is still one of finding a *set* of alternatives, albeit reduced in size, and so we can regard it as a special case of *a posteriori* optimization.

2.3 An overview of methods for generating a Pareto front

What methods can we use to build up an approximation to the true Pareto front (our goal as outlined above)? Leaving aside, for the moment, the finer details of the overall algorithm design, the initial question is simply: how can any solution be evaluated so that some form of heuristic search can be effected?

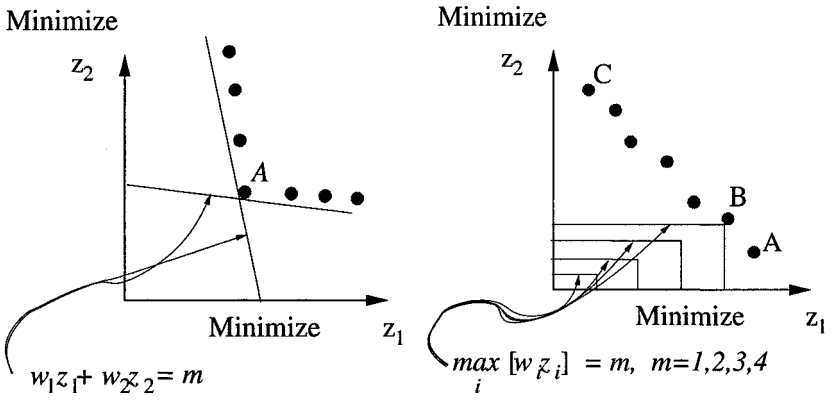

Fig. 3. The figure on the left shows a Pareto front where even a large change in the weights of a weighted sum scalarization would result in finding the same solution. On the right, the weighted Tchebycheff problem (equation 3) can find non-supported Pareto optima, as shown. Here, the reference point is taken as the origin

There are a great variety of answers possible. One large family of methods is to replace (1) with some parameterized single *scalarizing* function to minimize, such as a weighted sum of the objectives:

$$\text{minimize } \sum_{i=1}^{k} w_i.f_i(\mathbf{x}) \tag{2}$$

where we usually specify $\sum_k w_i = 1$ and $w_i \geq 0$, for $i \in 1..k$. Then, by varying the weighting parameters w_i in some systematic way, a representation of the PF can be built up. The weighted sum is only one possible method in this family of *scalarizing methods* and has some serious drawbacks. Only supported solutions—those on the convex hull of the PF—will be generated by minimizing the weighted sum. Furthermore, a small change in the weights can cause big changes in the objective vectors (see figure 2); while, on the other hand, very big changes in the weights may lead to the same or very similar vectors (figure 3, left). Other methods in this family that can generate the non-supported solutions are possible, e.g. the weighted Tchebycheff problem:

$$\text{minimize } max_{i \in 1..k}[\, w_i | f_i(\mathbf{x}) - z_i^* | \,] \tag{3}$$

where $\mathbf{z}^*$ is a reference point beyond the *ideal point*, i.e. each of its components is less than the minimum value possible on the corresponding objective. With such a reference point correctly specified, every Pareto optimal solution minimizes the function for some particular value of the weights. However, as with the weighted sum, in order to achieve an 'even sampling' of the Pareto front, care must be taken with how the weights are adjusted.

Other parameterized scalarizing methods include the epsilon-constraint method and achievement scalarizing functions: see [84] for further details.

Notice that these methods are suitable for exact algorithms, local searchers and so forth, since they effectively transform the problem back into a single-objective problem temporarily. So, for MAs, they may well be used as part of the overall algorithm.

With many metaheuristics, particularly traditional EAs, however, it is not necessary to have an explicit function to minimize, but only some means of estimating *relative* fitness (as in EA populations) or accepting/rejecting neighbour solutions (as in e.g., simulated annealing and tabu search). This opens the door to at least two other distinct approaches. One is to consider *alternately* one objective function then another; and there are various ways this could be organized (see section 3.5). The other approach is to use some form of relative ranking of solutions in terms of Pareto *dominance* (section 3.1 and 3.2). The latter is the most favoured approach in the EA community because it naturally suits population-based algorithms and avoids the necessity of specifying weights, normalizing objectives, and setting reference points.

In the last section, we discussed the reasons why we will restrict our working definition of MOO to be the problem of generating an approximation to the entire PF, ignoring methods that seek only a single solution. Following this, we went on to outline three general ways in which solutions could be evaluated in a search algorithm in order to effect optimization. In this section, we will expand greatly on this outline as we tour a host of metaheuristics for MOO. In addition, we will begin to appreciate two other related issues: how to build up the Pareto front during search (i.e. how to ensure a spread of solutions across it); and how memory of these solutions is organized to exploit them during search and/or to store them for presentation at the termination of the search process.

In the following we attempt a fairly broad survey of MOO algorithms in order to furnish the reader with a library of 'components' from which MAs could be constructed. We cluster different algorithms together in ad-hoc categories, as we review them.

2.4 Non-elitist EAs using dominance ranking

Goldberg in a short discussion in [44] suggested that multiple objectives could be handled in an EA using a ranking procedure to assign relative fitness to the individuals in a population, based on their relative Pareto dominance. The procedure, known as nondominated sorting, has become one of the bedrocks of the whole EMOO field. It is described and depicted in Figure 4. Although Goldberg did not implement it himself, it was not long before it gave rise to the popular NSGA [100]. The contemporaneous MOGA, of Fonseca and Fleming, [39] uses a slightly different ranking procedure based on counting the number of individuals that dominate each member of the population but otherwise the idea is very much the same.

Both NSGA and MOGA also employ *fitness sharing* [45], a procedure that reduces the effective fitness of an individual in relation to the number of other

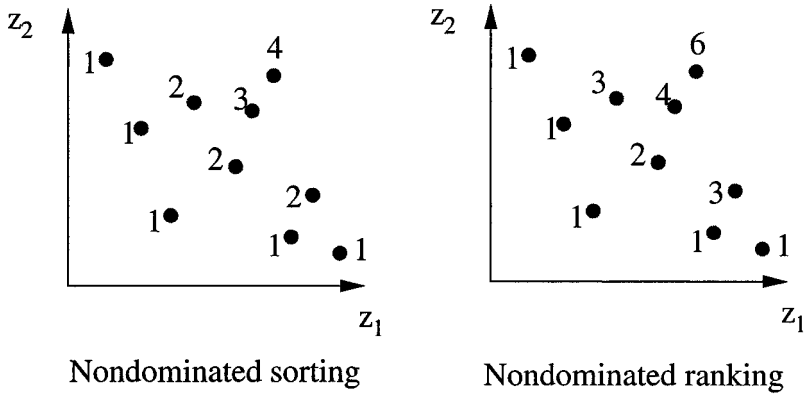

Fig. 4. On the left, individuals in a population are assigned dummy fitness values using Goldberg's nondominated sorting scheme. In this, successive iterations of the sorting procedure identify, and remove from further consideration, the nondominated set of solutions. A dummy fitness of 1 is assigned to the first set of solutions removed, and then fitness 2, and so on, 'peeling off' layers of mutually nondominated solutions. On the right, individuals in the same population are assigned fitness values using MOGA-style ranking, where fitness is 1+ the number of dominating solutions. Note, in both schemes, lower values are associated with greater fitness in the sense of reproductive opportunity or survival chances

individuals that occupy the same 'niche'. In MOO, the niche is often defined by the 'distance' of solutions to one another in the objective space, though parameter space niching may also be used. Sharing and other methods of niching have to be used in dominance-ranking MOEAs in order to encourage a spread of solutions in the objective space. Some objective-space niching methods are depicted schematically in Figure 5. Both NSGA and MOGA use similar methods to convert the shared fitness value to actual reproductive opportunity: a ranking-based selection.

The niched Pareto GA (NPGA) of Horn and Nafpliotis [53] uses, instead, tournament selection. In addition to the two individuals competing in each tournament, a sample of other individuals is used to estimate the dominance rank of the two individuals. In the case of a tie, again, fitness sharing was applied.

These EAs, NSGA, MOGA and NPGA, represent a trio that were tested and applied to more problems than any preceding algorithms for MOO, and pushed forward immensely the popularity and development of the evolutionary multiobjective optimization (EMOO) field. Most MOEAs today still use some form of dominance ranking of solutions, albeit often combined with *elitism*, and some form of niching to encourage diversity.

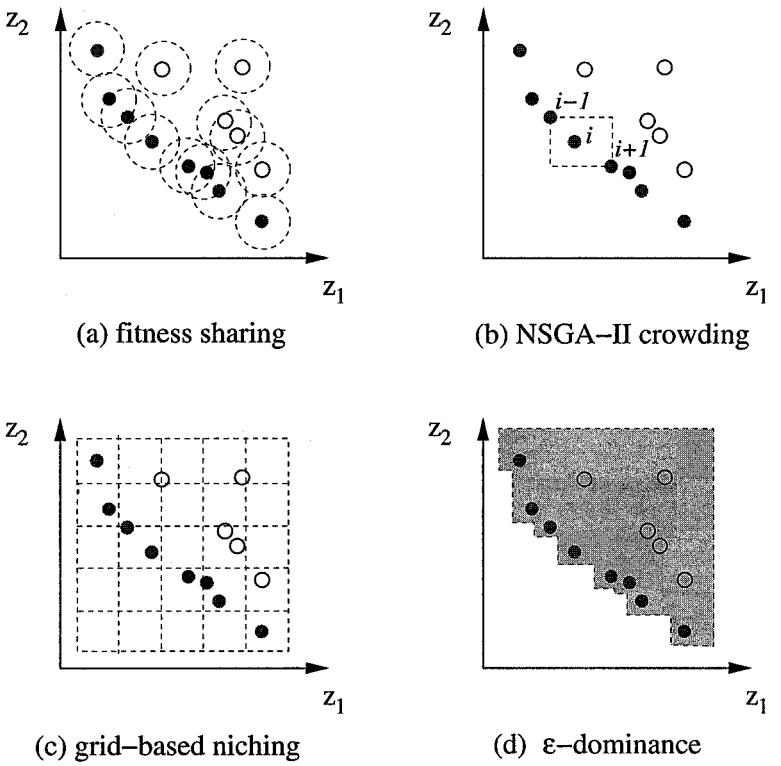

Fig. 5. Schematics depicting the different forms of niching used in various MOEAs to encourage diversity in the objective space; nondominated solutions are shown solid, and dominated ones are in outline. (a) *fitness sharing* (as used in NSGA and MOGA) reduces the fitness of an individual falling within another's niche (dashed circles), the radius being defined explicitly by a parameter. (b) NSGA-II *crowding* ranks solutions by measuring the distance of it's nearest nondominated neighbours, in each objective. (c) a grid is used in PAES, PESA and PESA-II, to estimate crowding: individuals in crowded grid regions have reduced chances of selection. (d) in ϵ-dominance archiving, a solution dominates a region just beyond itself, specified by the ϵ parameter so that the shaded region is forbidden — thus new nondominated solutions very nearby to those shown would not enter the archive

2.5 Elitist EAs using dominance ranking

Elitism in the EA terminology means the retention of good parents in the population from one generation to the next, to allow them to take part in selection and reproduction more than once and across generations.

The first multiobjective evolutionary algorithms employing elitism seem to have appeared at approximately the same time as MOGA, NSGA, and NPGA were put forward, around 1993-4 as reviewed in detail in [52]. In some elitist MOEAs, the strategy of elitism is combined with the maintenance of an 'external population' of solutions that are nondominated among all those found so far. Several early schemes are discussed in [112] but the first elitist MOEA paper to be published in the mainstream evolutionary computation literature was [94]. In this work, Parks and Miller describe a MOEA that maintains an 'archive' of nondominated solutions, similar to a store of all nondominated solutions evaluated, but limited in size: members of the main population only enter the archive if they are sufficiently dissimilar from any already stored. Reproductive selection takes parents from both the main population and the archive. The authors investigate the effects of different degrees of selection from each, and also different strategies for selecting from amongst the archive, including how long individuals have remained there.

At around the same time Zitzler and Thiele proposed what is to date one of the most popular of all MOEAs: the strength Pareto EA (SPEA) [113]. It uses two populations: an internal population, and an external population consisting of a limited number of nondominated solutions. In each generation, the external population is updated by two processes: addition of new nondominated individuals coming from the internal population (with removal of any solutions that consequently become dominated); and removal of solutions by objective-space clustering, to maintain a bound on the population's size. The new internal population is then generated by selection from the union of the two populations, and then by applying variation operators. The novelty, and perhaps the efficacy, of SPEA derives from the way the internal and external population interact in the fitness assignment step. In this, each external population member is first awarded a *strength*, proportional to the number of internal population members it dominates. Then each internal population member is assigned a dummy fitness based on the sum of the strengths of the external population members that dominate it. Binary tournament selection with replacement is used based on the dummy fitness/strengths of the combined populations. This fitness assignment strategy is a co-evolutionary approach between two distinct populations and its purpose is to bias selection towards individuals with a lower dominance rank *and* that inhabit relatively unpopulated 'niches'. The niches in SPEA are governed by the position of the nondominated individuals, and these are clustered so should themselves be well-distributed.

Numerous other elitist MOEAs exist in the literature, offering slightly differing ways of assigning fitness, choosing from a main population and an archive, and encouraging or preserving diversity. Regarding the latter, a trend towards self-adaptive niching (see Figure 5) has established itself with SPEA, PAES [73], NSGA-II [26], and PESA [20], amongst others, to avoid the necessity of setting niche sizes in the objective space, a problem with early algorithms such as MOGA and NSGA. Control of the degree of elitism has

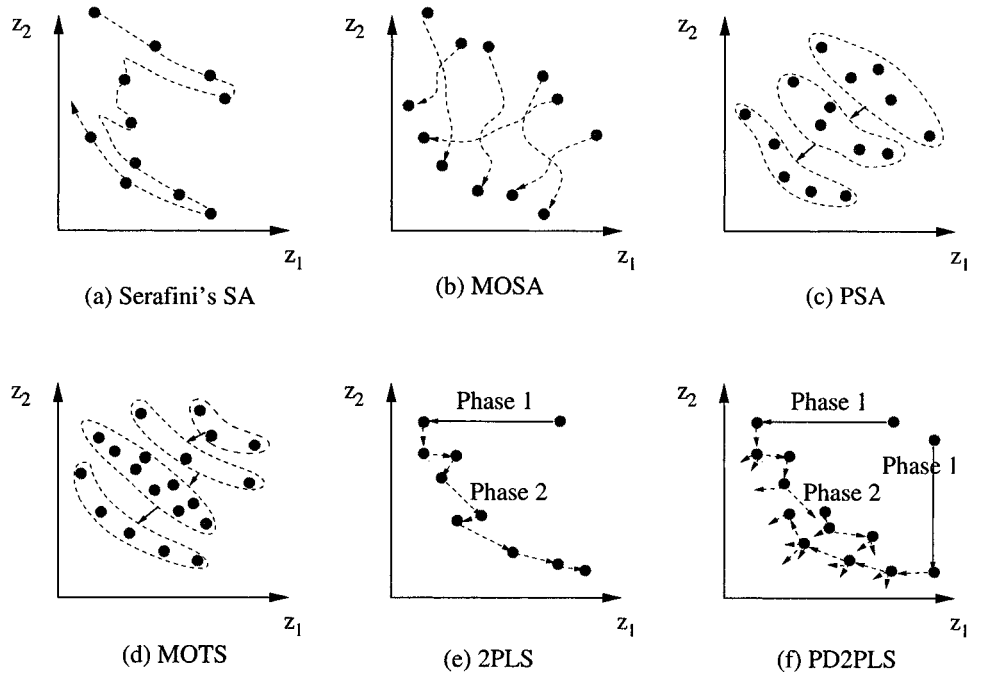

Fig. 6. Schematics depicting the different strategies employed by different local search metaheuristics, as described in section 3.3

also been investigated, e.g. in [27], and there has also been a trend towards lower computationally complexity, as evidenced by PAES, NSGA-II and the micro-GA [17]. More efficient data structures for ranking and niching available now [64] should make the current breed of elitist MOEAs a good starting point for designing good MAs for MOO.

2.6 Local search algorithms using scalarizing functions

One of the earliest papers on local-search metaheuristics for MOO is [99], which proposes and investigates modifications to simulated annealing in order to tackle the multiobjective case. A number of alternative acceptance criteria are considered, including those based on Pareto dominance, but the preferred strategy combines two weight-based scalarizing functions. In order to sample different Pareto optima during one run of the algorithm, the weights for each objective are slowly modified, at each fixed temperature, using a purely random (non-adaptive) scheme.

The MOSA method [107] follows Serafini regarding the modification of the SA acceptance function, but uses a different approach to building up the

approximation to the Pareto set. Where Serafini's approach varied the weights of the scalarizing function as the cooling occurred, MOSA method works by executing (effectively) separate runs of SA, each run using its own unique weight vector, and archiving all of the nondominated solutions found.

A population-based version of Serafini's SA is proposed and tested in [23]. The Pareto simulated annealing algorithm, PSA, performs each step of the SA algorithm 'in parallel' on each independent member of a population (N.B. the members are not in competition: there is no selection step), and each member carries with it its own weighting vector. Of particular note is the fact that the members of the population co-operate through an innovative adaptive scheme for setting their individual objective weights, in order to achieve a good distribution of solutions in the objective space. In this scheme, each member of the population continually adjusts its own weight vector to encourage it to move away from the nearest neighbour solution in the objective space.

These three SA algorithms, Serafini's SA, MOSA method, and PSA, illustrate three different ways to organize the building up of a Pareto front, respectively: (1) use a single solution and improve it, letting it drift up and down the PF via the use of randomly changing scalarizing weights; (2) use separate, independent runs and improve a single solution towards the PF, each run using a unique direction; (3) use a population of solutions and try to improve them all in parallel, at the same time encouraging them to spread out in the objective space. These alternatives are illustrated respectively in Figure 6 (a),(b),(c).

The idea of adaptively setting the weight vectors of individuals in a population, as used in PSA, is also used and extended in a tabu search algorithm, called MOTS [49]. In this, an initial population of points is improved in parallel, much as in PSA, but using a tabu search acceptance criterion. MOTS has another notable feature of particular relevance to MA design: it uses an adaptive population size based on the current nondominance rank of each member of the population. When the average of this rank is very low, it indicates that the members of the population are already well-spread (since few dominate each other), so the population size is increased in order to be able to cover more of the Pareto front. If the rank becomes too high this indicates that solutions are overlapping each other in objective space, and hence the population size is decreased—see figure 6(d).

Most recently, [90] describes a generic local search-based procedure for bi-objective problems, the two-phase local search (2PLS). In this approach the so-called 'first phase' applies local search to the problem, considering only one objective in isolation. When a good local optimum has been found, the 'second phase' begins. It uses the previous good solution as a starting solution for a new local search based on a scalarizing of the two objectives. Once a good solution has been found, the weights of scalarization are adjusted and the LS is again applied, again using the previous solution as a starting solution. Thus, a 'chain' of LS runs is applied, until a specified number of weights has been completed and the algorithm terminates (figure 6(e)). Depending

on the problem, the weights may be adjusted gradually or randomly. For the multiobjective TSP it is shown that gradual changes in the weights leads to good performance. In a slight variation to the algorithm, called the Pareto double two phase local search (PD2PLS), two first phases are used, one for each objective, and subsequently the best solution returned by each LS run is augmented using a search for nondominated solutions in its neighbourhood (figure 6(f)). This increases the number of nondominated solutions found by the algorithm with little overhead in time. Overall, the 2PLS and PD2PLS algorithms exhibit high performance on benchmark multiobjective combinatorial optimization problems, and are thus worthy contenders as subroutines for use within an MA for MOO, although versions for more than two objectives are needed.

2.7 Model-based searchers using dominance ranking

Model-based search is a name for a class of algorithms that employ some kind of statistical model of the distribution of remembered good solutions in order to generate new solutions. They can be seen as a development of EAs, in which recombination is replaced by a more statistically unbiased way of sampling from the components of known good solutions. Examples of model-based search algorithms are population-based incremental learning (PBIL), univariate distribution algorithms (UDAs), ant-colony optimization (ACO), Bayesian optimization algorithms (BOAs), and linkage-learning EAs. Recently a number of attempts at extending model-based search to the multiobjective case have been made, and like most MOEAs, they use the dominance ranking (see figure 4) to evaluate solution quality.

Straddling the middle-ground between a standard EA and a model-based search, the *messy genetic algorithm*, which attempts to learn explicit 'building blocks' for crossover to operate with, has been extended to the MOO case with the MOMGA and MOMGA-II algorithms [108, 115].

A step further away from standard, recognisable EAs, are algorithms that replace recombination altogether by using instead an explicit probability distribution over solution components, in order to generate new solutions. Several different attempts have been made at adapting Bayesian optimization algorithms (BOAs) and similar variants, to the multiobjective case [65, 80, 98, 106]. In the models proposed in [106], it is found that a factorization based on clusters in the objective space is necessary to obtain a good spread across the Pareto front. This results in an algorithm that is quite similar to the population-based ACO [47], described below, except that here the model is based only on the current population and not on a selection from a store of all nondominated solutions. The approach of [80] is a little different: instead of a mixture of clustered univariate distributions, a binary decision tree is used to model the conditional probabilities of good solution components. In order to encourage this model to generate sufficient diversity in the objective space, the selection step is based on ϵ-dominance [82] (see figure 5),

whereby solutions that are very similar tend to ϵ-dominate each other and will not be selected.

Ant colony optimization [30], is an agent-based search paradigm, particularly suited for constrained combinatorial optimization. Briefly, in this approach, candidate solutions are constructed 'component by component' by the choices made by 'ants' as they walk over a solution construction graph. At each step of a solution construction, the components available for the ants to select have associated with them a particular desirability, which biases the selection. This bias is mediated through the concentration of pheromone on the nodes or edges of the construction graph. In the usual implementations of ACO, the initially random pheromone levels change gradually via two processes: depositing of pheromone on the components making up a very good solution whenever one is found, and evaporation of pheromone, as a forgetting mechanism to remove the influence of older solutions. In population-based ACO, no evaporation is used, and instead a population of good solutions is always stored. Whenever a solution in the population is replaced by a new one, the pheromone trails associated with the old one are entirely removed from the construction graph, and the new member of the population deposits its pheromone instead. In [47], population-based ACO is adapted to the multiobjective case. This is achieved by making use of a store of all nondominated solutions found, and periodically choosing a subset of this to act as a temporary population. Promotion of diversity in the objective space is achieved in two ways: (1) the members of a temporary population are selected from the nondominated set based on their proximity to one another in the objective space (so there is a kind of restricted-mating or island-model effect); and (2) each objective has its own pheromone and the selection of components is governed by a weighted sum over the different pheromone levels—the weights being determined by the location, in objective space, of the current temporary population, relative to the entire nondominated set.

2.8 Algorithms using alternating objective functions

Schaffer is widely regarded as having started the field of evolutionary multiobjective optimization with his seminal paper on the vector evaluated genetic algorithm (VEGA) [97]. This was a true attempt at the evolution of multiple nondominated solutions concurrently in a single EA run, and the strategy was aimed at treating possibly non-commensurable objectives. Thus, aggregation of objectives was ruled out in favour of a selection procedure that treated each objective separately and alternately. As explained in [40], the approach is, however, *effectively* assigning reproduction opportunities (fitness) as a weighted linear function of the objective functions, albeit it implicitly adapts the weighting to favour the objective which is 'lagging' behind. This behaviour means that on problems with concave Pareto fronts, 'speciation' occurs, meaning that only solutions which do well on a single objective are found, while compromise or middling solutions do not tend to survive. Another early

approach, this time using evolution strategies (ESs) as the basis, was proposed by Kursawe [78]. The paper included some interesting early ideas about how to deal with non-commensurable objectives but the algorithm proposed has not been tested thoroughly to date.

Nearly ten years younger than the latter, [102], describes one of the first distributed EAs for MOO. It employs three separate but interacting populations: a main population and two islands, with the main population accepting immigrants from the islands. The performance of three strategies were compared. One strategy is to use homogeneous populations, each evolving individuals using the dominance ranking for fitness assignment. The second is to use heterogeneous islands, each evolving individuals to optimize a different objective, while the main population is still evolved using dominance ranking. The third is the same as the second but restarts are additionally used in the island populations. Testing on a number of scheduling problems revealed the latter to be consistently and significantly the most effective and efficient of the three strategies.

Gambardella*et al.* use a similar kind of heterogeneous, co-operative approach in their ant-colony optimization algorithm for a vehicle routing problem [42]. The problem tackled has two objectives: to minimize the number of vehicles needed to visit a set of customers with particular time window constraints; and to minimize the total time to complete the visits. To achieve this, two separate ant colonies work pseudo-independently and in parallel. Starting from a heuristically generated feasible solution, one colony attempts to minimize the number of constraint violations when one fewer vehicle is used than in the current best solution, while the other colony attempts to reduce the total time, given the current best number of vehicles. Feasible improvements made by either colony are used to update the current best solution (which is used by both colonies to direct construction of candidate solutions). In the case that the colony using one fewer vehicles finds a feasible solution, both ant colonies are restarted from scratch, with the reduced number of vehicles.

2.9 Other approaches

One MOO approach which stands very much on its own is a method proposed in [37]. The originality of the approach lies in the way the whole multiobjective optimization problem is viewed. In every other approach outlined above, whether it be population-based, model-based, or a local search, it is individual solutions that are evaluated, and the fitter ones somehow utilised. By contrast, [37] proposes evaluating the whole current population of solutions *in toto* and using this scalar quantity in an acceptance function. For example, simulated annealing in this scheme would work by applying some measure (and Fleischer proposes the Lebesgue integral of the dominated region – see figure 7) over a population of current solutions. When a neighbour solution of one of the population is generated, it is accepted modulo the change in the Lebesgue measure of the *whole* population. Fleischer points out that since the

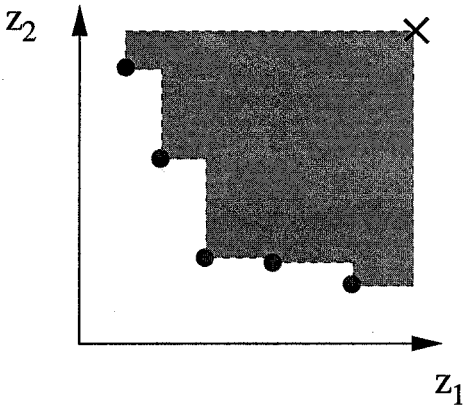

Fig. 7. The Lebesgue measure (or $\mathcal{S}$ metric) of a nondominated approximation set is a measure of the hypervolume dominated by it (shaded region), with respect to some bounding point (here shown by an X). The maximum of the Lebesgue measure corresponds to the Pareto front

maximum of the Lebesgue integral is the Pareto optimal set (provided the number of solutions is large enough), a simulated annealing (for example) optimizing this measure provably converges in probability to the Pareto optimal set.

3 Going Further: Issues and Methods

We have seen in the last section a variety of metaheuristic approaches to MOO, illustrating some of the basic principles of how to assign fitness and maintain diverse 'populations' of solutions. These are the basic pre-requisites for MAs for MOO, however a number of further issues present themselves. In this section we briefly discuss the current thinking on some of these other issues.

3.1 Performance measures in MOO

If one is developing or using an algorithm for optimization it almost goes without saying that there should be some way to measure its performance. Indeed, if we are to compare algorithms and improve them we really must first be able to define some means of assessing them. In single-objective optimization it is a relatively simple case of measuring the quality of solution obtained in fixed time, or alternatively the time taken to obtain fixed quality

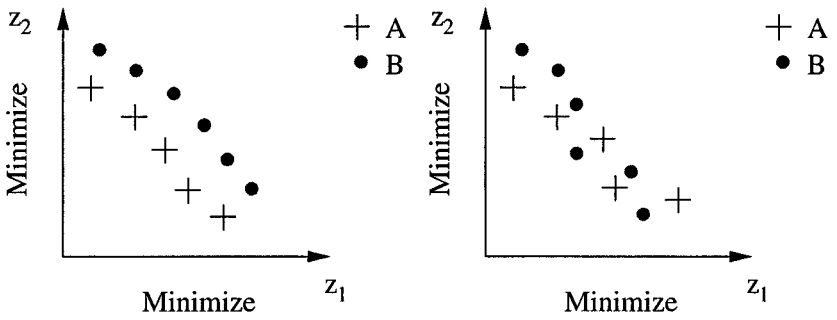

Fig. 8. On the left, two sets A and B, where A outperforms B, since every vector in B is dominated by at least one in A. On the right, two sets that are incomparable—neither is better under the minimal assumptions of Pareto optimization

– and quality and time can themselves be defined unequivocally in some convenient way. In MOO the situation is the same regarding the time aspect of performance assessment but the quality aspect is clearly more difficult. Recall that the standard goal of MOO (as far as we are concerned) is to approximate the true Pareto optimal set, and hence the outcome of the search is not one best solution, but a set of solutions, each of which has not one, but multiple dimensions of quality. We call these approximation sets, and it is clear that approximation sets cannot be *totally* ordered by quality, (see figure 8), if we remain loyal to the minimal assumptions of Pareto optimization. Nonetheless, a *partial* order of all approximation sets does exist, so it is possible to say that one set is better than another for some pairs, while others are *incomparable.*

The partial ordering of approximation sets is sometimes unsatisfying because it, of itself, does not enable an approximation set to be evaluated in isolation. For this reason, practitioners sometimes (often implicitly) adopt an ad hoc definition of a 'good approximation set' as one exhibiting one or more of: *proximity* to the true PF; *extent* in the objective space; and a good or even *distribution*—and use measures for evaluating these properties. The problem with such an approach (if not done with great care and thought) is that these measures can conflict utterly with the stated goal of approximating the PF. This problem is illustrated in Figure 9.

If one wants to really do Pareto optimization, and needs a unary measure of approximation set quality, the fact that there is a true partial ordering of all approximation sets (under Pareto optimization assumptions) demands that good or reliable measures of quality respect this ordering in some way. Using this fact, it is possible to assess how useful and reliable are different potential measures of approximation set quality. If a measure can judge an approximation set A to be better than B, when the converse is true, for some pair of sets A and B, then the measure is, in a sense, unreliable and fairly useless. On the other hand if a measure never states that A is better

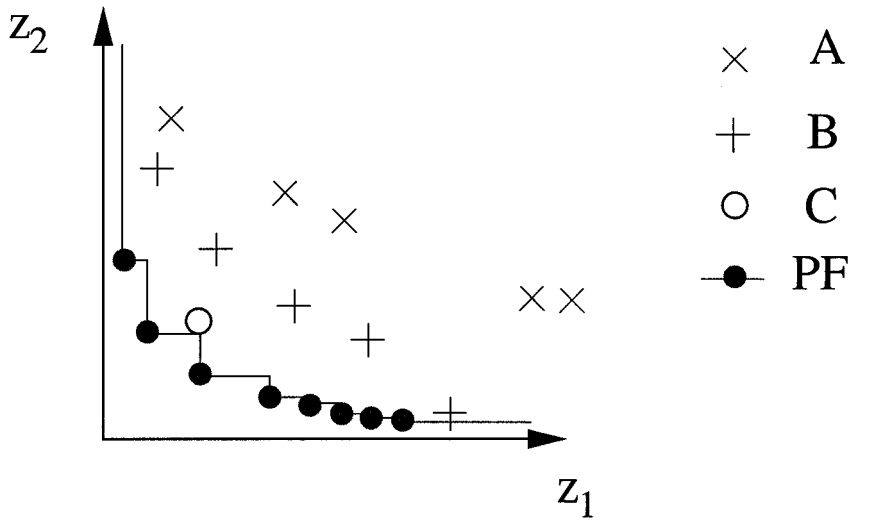

Fig. 9. A Pareto front and three approximation sets, A, B, and C. Depending on the measure used A, B, or C might be considered the best, and even better than the PF! If a measure of 'well-distributednes' is used B is best, better even than the PF. If a measure of proximity to the PF is used, C is best, even though B is just as close in parts, and if extent in objective space is measured, A is best

than B when the reverse is true, then it may be of some use, even if it does not detect all positive cases. More useful still, is if it always detects positive cases correctly but sometimes judges one set better when they are, in fact, incomparable. The ideal situation is when a measure always detects A better than B when it is the case, and never gives a false positive, when it is not.

A plethora of different measures for overall or specific aspects of MOO performance are described in the EMOO literature but it is not until relatively recently that some researchers have begun to critically assess them. Most notably, Zitzler *et al.* [114] give an extensive treatment of performance measures in MOO, using a framework that formalizes the notion of respecting the true partial ordering of approximation sets, as described above. A key result of [114] is that no unary measure (i.e. one taking just one approximation set as input, and returning a number or vector of numbers) whatever, including any finite combination of unary measures, can detect reliably when one approximation set is better than another, without giving false positives. Such results underline the necessity of thinking very hard before selecting measures to evaluate performance. For earlier work on the same issue, see also [50, 68, 72].

3.2 Archiving, multi-populations, and convergence

In the tour of metaheuristics of section 3 we saw some examples of algorithms using secondary populations, archives, and/or populations of non-fixed size. The use of such mechanisms seems to be a necessary element of more advanced methods for MOO, which aim to build up and store a good approximation to the PF. Some of the options for incorporating these elements within existing algorithms are summarised below:

- Use a main population only: the population is the store (NPGA [53], NSGA [100], NSGA-II [26])
- Use a single-point local search, but keep separately a bounded-sized archive of nondominated points found (PAES [73])
- Use a main and a secondary population - both of fixed size (SPEA [113], PESA [20])
- Use a main population of fixed size and an archive of unbounded size [36] (RD-MOGLS [60])
- Use multiple populations as in an island model [66]
- Have a dynamic main population size [105] (MOTS [50])

With the use of an (unbounded) archive of solutions, an algorithm can potentially converge to the (entire) Pareto front. Thus, convergence proofs for MOEAs now exist in the literature: Rudolph [96] proved convergence to the Pareto front (that is at least one Pareto optimal solution) for some simple multiobjective EAs, and it is possible to prove that the entire Pareto front can be enumerated provided an unbounded archive is available. More realistically, archives should be bounded in some way. A number of more recent papers have been written regarding what can be proved with respect to the convergence properties of such bounded archives [71, 70, 74, 82, 81]. Research in this area is still needed and the issue of which solutions to store during the search in order to converge towards a 'good approximation set' remains an open question.

3.3 Test functions, problems, and landscapes

Early test functions in the EMOO literature comprised a number of ad hoc, low-dimensional functions, enabling a proof-of-concept for early MOEAs, but nothing more. A large step toward a more scientific approach was taken with the introduction, by Deb [25], of a framework for constructing functions with identifiable features such as concave, discontinuous, and non-uniform PFs, local optima, and deception. A suite of six 2-objective functions derived from these became popular for some time, despite some drawbacks. Deb later extended the framework in [29] to allow scalable functions of any number of objectives to be generated and this is becoming more popular for testing now.

There is still a lack of understanding of the relationships between the properties of problems, their 'landscapes', and strategies for search. A couple of studies that have begun investigating this with respect to combinatorial

problems are [9] and [69]. [9] investigates 'global convexity' in multiobjective TSP problems and finds that optima nearby to each other on the PF are quite similar, a finding that suggests restricted mating in MOEAs and strategies for chaining local searches along the PF, like the 2PLS [90], would be effective on this problem. [69] introduces a tunable multiobjective version of the QAP problem and proposes some techniques for characterising the landscapes of instances of this problem. The latter is ongoing work.

3.4 Not quite Pareto

In much of the above discussion we have explicitly stated that finding a good approximation to the entire Pareto front is the goal of MOO, as far as we are concerned. Nonetheless, several situations arise when finding the whole Pareto front may not be desirable, and yet finding a single solution, as would be obtained by transforming the problem into a single objective, would not be adequate either. In particular, when the number of objectives is much above two or three, the size of the Pareto optimal set may be very large, necessitating a more restrictive notion of optimal. In these cases, some kind of 'middle ground' may be the best option, in which some Pareto optima may be treated as more desirable than others. One of the seminal papers on non-Pareto approaches is [7], which proposes a number of alternative ranking policies for use in EA selection. Other more recent policies are described in [10, 34], and in [31] where the concept of the order relation, 'favour', is introduced. Where more explicit preferences of a decision maker are available more advanced methods may be used, as in, for example, [22].

4 MAs for MOO: the fossil record

The extensive array of existing metaheuristics, issues and methods reviewed in the sections above gives a richer basis from which to design new MAs than do the existing MAs for MOO themselves. Nonetheless, before outlining some principles and ideas for new MAs, it is worth reviewing the few multiobjective MAs described in the current literature.

Arguably, it is just three separate groups of authors that are responsible for much of the small multiobjective MA literature, each group having written several papers. A small number of others have published more isolated works and these tend to be application-based rather than aiming at developing general algorithms. The three main groups are: Ishibuchi and Murata, who proposed a 'multiobjective genetic local search' (MOGLS) [54] algorithm in 1996; Jaszkiewicz, who proposed an algorithm initially called RD-MOGLS for 'random directions' MOGLS [60], and a slight variant called the Pareto memetic algorithm (PMA) [61]; and Knowles and Corne, who developed an algorithm called M-PAES [67]. In all of these algorithms, the basic idea is simple: a local search is applied to every new offspring generated (by crossover

or mutation), and the improved offspring then competes with the population for survival to the next generation (Lamarckianism). In all cases, only one local search operator is available and there has been no work on mechanisms for deciding whether or not to apply a local search to an offspring. The algorithms of Ishibuchi and Murata and Jaszkiewicz are quite similar in other respects too: both use randomly-drawn scalarizing functions to assign fitness for parent selection and in the local search. The algorithm of Jaszkiewicz uses an unbounded 'current set' of solutions, CS, and from this selects a small 'temporary population', TP that comprises the best solutions on the incumbent scalarizing function. It is then TP that is used to generate offspring by crossover. Some results put forward by Jaszkiewicz suggest that scalarizing functions are particularly better at encouraging diversity than dominance ranking methods used in most EAs. Ishibuchi and Murata have also made a number of interesting studies on their algorithm over the years, investigating restricted mating and other innovations, and have tested it on several problems [55, 86, 87, 88, 58, 57, 56, 89]. Knowles and Corne's M-PAES algorithm is quite different in at least one respect from the other two: it does not use scalarizing functions at all, either in the local search or the parental selection, employing instead a form of Pareto ranking based on comparing solutions to an archive of nondominated solutions. This may perhaps make it slower when very fast local search heuristics are available because the comparison of solutions takes longer to operate than applying a scalar acceptance function. On the other hand, whereas the MOGLS algorithms will discard newly generated nondominated solutions if they are poor on the incumbent scalarizing function, this will not happen in M-PAES, making it potentially more parsimonious of function evaluations—an advantage when these are more costly.

Of the more isolated papers, a few stand out for their interesting ideas or applications. In [43] the idea of using supported solutions (figure 9) to seed an EA is proposed. That is, on problems where some exact algorithm for computing supported Pareto optima is available, [43] proposes a two-phase hybrid approach where the exact algorithm is applied first, then an EA is used to search for the non-supported Pareto optima, which cannot be found using the exact heuristic.

Another kind of two-phase approach is described in [103]. The proposed procedure is as follows: run an MOEA for a fixed number of generations; then for each Pareto optimal solution, compute the neighbourhood and store any nondominated solutions found; update the list of PO solutions and again recompute all the neighbourhoods; iterate the procedure until no improvement occurs.

Similarly to [103], [27] proposes to run an EA (NSGA-II) and then apply local search afterwards to improve the Pareto optimal set. To do this, the authors apply a local search using a weighted sum of objectives. The weights used are computed for each solution based on its location in the Pareto front such that the direction of improvement is roughly in the direction perpendic-

ular to the PF. Nondominated solutions are then identified and clustering is finally applied to reduce the number of solutions returned.

Finally, worth mentioning because the results on a well-known application problem are apparently good, is [1]. This paper introduces a hybrid of differential evolution [101] and backpropagation learning in order to evolve both the architecture and weights of an artificial neural network. Two objectives are minimized, the summed squared error in training, and the number of neural units in the network. Abbass reports good reports on the Australian Credit Card and Diabetes Data sets.

5 Recommendations for MA design and practice

In the previous sections we have reviewed current MOO practices: we revisited a swathe of metaheuristics, considered some of the most salient issues and results, and looked briefly at some existing MAs. We now consider how we should draw on this background to build a more 'memetic' MA for MOO.

In recent years, Moscato and Krasnogor have provided a guiding manifesto for putting the 'memetic' back in memetic algorithms [77, 85] advocating, in particular, the use of multiple memes: memeplexes. These are collections of ways of learning or adapting which can be transmitted at different levels and through different processes. For example, multiple local searches, multiple recombination operators, and so on could co-exist in a single algorithm, that then learns, at both the individual and the population level, which operators to use, and when, depending on the monitoring of internal processes at the level of the individual or population. The MAs that we have seen in the MOO literature to date are relatively poor images of these 'fully-fledged' MAs.

In Algorithm 1, we put forward a simple framework that could serve as a guide for making a more memetic MA for MOO. In line 1, a population P of solutions is initialized. As usual, this procedure may be simply random or it may employ some heuristic(s). Line 2 sets the archive A to the nondominated solutions from P. Thereafter, the main loop of the MA begins. Line 4 sets up an inner loop in which a stagnation criterion is checked. This should be based on some memeplex which monitors progress in diversity, proximity, and/or some other criteria. Lines 5–9 give a very high level description of the update of the population and archive. Five different 'schedulers' are employed, basically corresponding to mating selection, reproduction, lifetime learning, survival selection, and update of the archive, respectively. Each scheduler chooses from a memeplex of operators, based on estimates of the current success of those operators. E.g., in line 5, SelectFrom is the operation of mating selection, the domain of which is the union of the population and archive, and co-domain is a child population C; the selection is controlled by the scheduler, sel_sched, which uses a success measure, succ, to choose one operator from the set, *SEL*, of currently available operators for selection. Notice that P and A are potentially of variable size, in this scheme. In line 11, the population P is

```
Multi-Objective MA():
Begin
  P := Initialize(P);
  A := Nondom(P);
  While ( stop_criterion not satisfied ) Do
    While ( stagnation_criterion not satisfied ) Do
      C := SelectFrom(P ∪ A, sel_sched(succ(SEL)));
      C' := Vary(C, var_sched(succ(VAR)));
      C'' := LocalSearch(C', ls_sched(succ(LS)));
      P := Replace(P ∪ C'', rep_sched(succ(REP)));
      A := Reduce(Nondom(A ∪ P), red_sched(succ(RED)));
    endDo
    P := RandomImmigrants(P, imm_sched(succ(IMM)));
  endDo
  return (A);
End.
```

Fig. 10. Candidate MA framework for MOO

updated using some immigration policy to release it from stagnation. The archive of nondominated solutions is returned in line 13.

The framework proposed is rather broad and actually instantiating it requires us to consider how we should resolve many choices, including those considered in the following sections, at the very least. Table 1 summarises some of the MA elements/configuration choices to consider.

5.1 Desired outcomes and prevailing conditions

As in any other optimization scenario, we should know at the outset what is a desirable outcome, how this can be measured, and what are the prevailing conditions under which the search is going to take place.

One important factor in MOO is knowing how many solutions are desired. The answer could be as many as possible, an exact number, or could be expressed in terms of some resolution at which the PF is sampled. These considerations might affect different options for storing the nondominated solutions (see Table 1, question 1).

The dimensionality of the objective space is another important factor and how this is going to be dealt with. If there are only two or three objectives then there is some evidence that dominance-ranking-based selection methods may be the most appropriate, assuming a good approximation to the Pareto front is desired, with no particular preference for either diversity or proximity. On the other hand, if the number of objectives is high, Pareto selection may be problematic, because many solutions will be incomparable. There

Table 1. Some suggestions for configuring an MA design

Question	Answer	Choices
1. How many solutions are desired?		
	precisely N:	Use Lebesgue archiving [75] or adaptive grid archiving [70]
	as many as possible:	Use an unbounded archive [36]
	require ϵ-approx set:	Use ϵ-Pareto archiving [82]
2. Is diversity more or less important than proximity?		
	More:	Use scalarizing functions; optimize diversity only
	Less:	Use strong elitist selection; dominance ranking approach;
	No preference:	Combine Pareto approach with scalarizing methods; monitor progress using an overall unary measure like the $\mathcal{S}$ metric
3. What is the dimensionality of the objective space?		
	1-d:	Consider 'multi-objectivizing' [76, 63]
	2-d or 3-d:	Use Pareto-ranking approaches
	4-d+:	Use the order relation *favour* [31], or preference methods [22], to reduce the number of effective optima; consider aggregating correlated objectives
4. How long does function evaluation take?		
	Minutes-to-days:	Use Bayesian approach [80] , or other computationally intensive model-based methods
	Seconds:	Use self-adaptation, other medium-overhead methods
	Microseconds:	Rely on fast LS strategies [90]
5. Is the true Pareto front known?		
	Yes:	Use epsilon-measure to compute progress/measure overall performance [114]
	No:	Use $\mathcal{S}$ measure to compute progress/measure overall performance
6. Are supported solutions available?		
	Yes:	Seed the MA with them and try to find the non-supported solutions [43]

are various alternatives to consider: using the relation *favour* [31], instead of component-wise order; using some other aggregating methods as proposed in [7]; or, actually aggregating some of the objectives together following a correlation/mutual information analysis.

5.2 Methods for monitoring progress

The MA framework proposed above requires that operators and procedures be selected based on their current success rates. These, in turn, must be estimated by some notion of progress. How should this progress be measured? Deb has proposed a number of running time metrics in [28] and Zitzler has advocated using the $\mathcal{S}$ metric [112] (see figure 7) to detect convergence. We have not seen much in the way of statistics for detecting or measuring the success of particular operators so far but these could be adapted from similar measures used in EAs.

5.3 System- and self-adaptation

Adaptation of mutation rates, crossover probabilities, and so forth is a topic that has received significant attention in the EAs literature over the years (see [33] for an extensive review). By comparison, the topic of self-adaptation in MOO, is surprisingly under-developed. Where it has been used, as we saw earlier, is in the control of the search 'direction' in the objective space: i.e. to direct the search towards sparsely populated areas of the Pareto front. Thus, it is usually some kind of weighting vector adaptation. The potential in MOO for self-adaptation is large, however, and should be part of any 'real' memetic approach. The adaptation of selection pressure/elitism maybe of particular importance, since we would expect that the different stages of building up a Pareto front might demand more or less aggressive searches. Getting to a local Pareto front quickly may demand aggression, whereas stagnation there might suggest decreasing the selection pressure in order to spread out along it, or hopefully find a route to a better front.

Some attempts have been made at adapting population size to the size of the Pareto front – whether that be the archive population or the main one. E.g., as we saw in MOTS, the mutual dominance of the population was used to adapt the size of the population. In the work of Laumanns, it is by setting a desired level of approximation that the archive's size is controlled, so that an appropriate number of solutions is maintained. These methods seem to be going in the right direction, as the use of a fixed population size, when trying to search efficiently a multi-dimensional objective space would seem to be too restrictive.

5.4 Controlling the overall search

Let us assume that we are interested in maximizing the rate at which the S measure increases – that is our gold standard of progress. Then, we could have

a number of overall search strategies competing with each other in some form of bidding mechanism, where they each have wealth in proportion to a record of the prior rates of progress they achieved when in control of the search. We could have, for example, the following two search strategies:

1. A PLS-like strategy that applies a local search repeatedly in one direction, using an aggregation function, until some convergence criterion is fulfilled. After this, a nearby weight is chosen and the same solution is once again improved. All nondominated solutions are stored in an archive.
2. A PESA-like strategy in which a whole front of nondominated solutions is used to generate new solutions, generation by generation, via recombination and variation, with selection based on crowding.

One strategy could be chosen at random to start with. After each 'generation' the strategy of choice could be reviewed. However, changing a strategy could be tabu for some time immediately after a change, in order to give it a chance and for decent statistics on it to be collected. Noticeable drops in progress rate could invoke a change in the current strategy in use.

Much work is needed to investigate if advanced 'multi-meme' approaches like this illustrative example really could provide robustness over different types of landscapes arising from different problems, or indeed within a single problem. It is not clear, even from the existing single objective literature, that this kind of high-level adaptation is really beneficial, but the time is perhaps overdue for us to try and find out.

6 Future Prospects

What does the future of multiobjective MAs hold and what are the most promising avenues to investigate now? In this article we have tried to distil a rich soup of ideas from the ever-growing literature on multiobjective metaheuristics, and a little on MAs, in order to provide some basis for the generation of new, more advanced algorithms. Many of the basics will probably remain the same: solutions will be evaluated by a combination of Pareto ranking-type methods and scalarizing methods; diversity will be encouraged using niching and crowding in parameter and objective space, and by the controlled use of different weights in the scalarizing functions. However, there is great scope for building more advanced and more memetic algorithms. In particular, it seems that the need, unique to MOO, to obtain and maintain a diverse pool of different solutions, suggests that such things as adaptive population sizes, multi-populations, and combinations of local and global search are especially relevant.

Expanding on this, and looking into the near future, we see that there is potential for more investigation as to the effects of restricted mating schemes, and how the success of these relates to features of the underlying problem

and/or multiobjective landscapes. For problems with a large number of objectives, new non-Pareto methods for ranking solutions need more investigation, as do methods for analysing correlations between objectives, perhaps to combine some objectives together; and conversely, we have seen some evidence in the recent literature [76, 63] that even single-objective problems may be tackled more effectively using multiobjective methods – work which merits further attention.

We have also provided in this article a glimpse of the different possible routes to building up a Pareto front employed by different multiobjective algorithms and have hinted at ways that these different overall strategies could be combined together in self-adaptive strategies that are sensitive to the progress being made in the search. This area, we think, is most promising.

New, advanced data structures for the storage and retrieval of Pareto optima [64] may offer increased speed of MAs and EAs which will, if developed further, enable exact solutions to be found even in relatively large solution spaces, provided fast local searches and evaluation functions are available.

And at the other end of the spectrum, where the evaluation of a solution takes a relatively long time, the recent advanced methods in model-based search promise a more principled way of sampling the search space. We have yet to see how these could be combined with local searches and other heuristics to build advanced MAs for these tough problems but the future is certainly exciting.

References

1. Hussein A. Abbass. A Memetic Pareto Evolutionary Approach to Artificial Neural Networks. In *The Australian Joint Conference on Artificial Intelligence*, pages 1–12, Adelaide, Australia, December 2001. Springer. Lecture Notes in Artificial Intelligence Vol. 2256.
2. M. F. Abbod, D. A. Linkens, and M. Mahfouf. Multi-Objective Genetic Optimization for Self-Organizing Fuzzy Logic Control. In *Proceedings of UKACC Control'98*, pages 1575–1580, University of Wales Swansea, UK, september 1998. IEE.
3. M.A. Abido. A new multiobjective evolutionary algorithm for environmental/economic power dispatch. In *Power Engineering Society Summer Meeting*, volume 2, pages 1263–1268. IEEE, 2001.
4. Antonino Augugliaro, Luigi Dusonchet, and Eleonora Riva Sanseverino. Evolving non-dominated solutions in multiobjective service restoration for automated distribution networks. *Electric Power Systems Research*, 59(3):185–195, October 2001.
5. Richard Balling and Scott Wilson. The Maximim Fitness Function for Multiobjective Evolutionary Computation: Application to City Planning. In Lee Spector, Erik D. Goodman, Annie Wu, W.B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Com-*

putation Conference (GECCO'2001), pages 1079–1084, San Francisco, California, 2001. Morgan Kaufmann Publishers.
6. Benjamín Barán, José Vallejos, Rodrigo Ramos, and Ubaldo Fernández. Reactive Power Compensation using A Multi-objective Evolutionary Algorithm. In *IEEE Porto Power Tech Proceedings*, volume 2, pages 6–11, Porto, Portugal, September 2001. IEEE.
7. P. J. Bentley and J. P. Wakefield. Finding Acceptable Solutions in the Pareto-Optimal Range using Multiobjective Genetic Algorithms. In P. K. Chawdhry, R. Roy, and R. K. Pant, editors, *Soft Computing in Engineering Design and Manufacturing*, Part 5, pages 231–240, London, June 1997. Springer Verlag London Limited. (Presented at the 2nd On-line World Conference on Soft Computing in Design and Manufacturing (WSC2)).
8. To Thanh Binh and Urlich Korn. Multicriteria control system design using an intelligent evolution strategy with dynamical constraints boundaries. In *Proceedings of the Conference for Control of Industrial Systems (CIS'97)*, volume 2, pages 242–247, Belfort, France, 1997.
9. Pedro Castro Borges and Michael Pilegaard Hansen. A basis for future successes in multiobjective combinatorial optimization. Technical Report IMM-REP-1998-8, Institute of Mathematical Modelling, Technical University of Denmark, March 1998.
10. Jürgen Branke, Thomas Kaußler, and Harmut Schmeck. Guidance in Evolutionary Multi-Objective Optimization. *Advances in Engineering Software*, 32:499–507, 2001.
11. Larry Bull and Matt Studley. Considerations of Multiple Objectives in Neural Learning Classifier Systems. In Juan Julián Merelo Guervós, Panagiotis Adamidis, Hans-Georg Beyer, José-Luis Fernández-Villaca nas, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature—PPSN VII*, pages 549–557, Granada, Spain, September 2002. Springer-Verlag. Lecture Notes in Computer Science No. 2439.
12. Edmund K. Burke, Patrick De Causmaecker, Sanja Petrovic, and Greet Vanden Berghe. A Multi Criteria Meta-heuristic Approach to Nurse Rostering. In *Congress on Evolutionary Computation (CEC'2002)*, volume 2, pages 1197–1202, Piscataway, New Jersey, May 2002. IEEE Service Center.
13. Donald H. Burn and Jeanne S. Yullanti. Waste-Load Allocation using Genetic Algorithms. *Journal of Water Resources Planning and Management*, 127(2):121–129, March-April 2001.
14. H. W. Chen and Ni-Bin Chang. Water pollution control in the river basin by fuzzy genetic algorithm-based multiobjective programming modeling. *Water Science and Technology*, 37(8):55–63, 1998.
15. Scott E. Cieniawski, J. W. Eheart, and S. Ranjithan. Using Genetic Algorithms to Solve a Multiobjective Groundwater Monitoring Problem. *Water Resources Research*, 31(2):399–409, February 1995.
16. Carlos A. Coello Coello and Carlos E. Mariano Romero. Evolutionary Algorithms and Multiple Objective Optimization. In Matthias Ehrgott and Xavier Gandibleux, editors, *Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys*, pages 277–331. Kluwer Academic Publishers, Boston, 2002.
17. Carlos A. Coello Coello and Gregorio Toscano Pulido. A Micro-Genetic Algorithm for Multiobjective Optimization. In Eckart Zitzler, Kalyanmoy Deb,

Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 126–140. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.
18. D. Corne, M. Dorigo, and F. Glover. *New Ideas in Optimization*. McGraw-Hill, London, UK, 1999.
19. David W. Corne, Kalyanmoy Deb, Peter J. Fleming, and Joshua D. Knowles. The Good of the Many Outweights the Good of the One: Evolutionary Multi-Objective Optimization. *Connections. The Newsletter of the IEEE Neural Networks Society*, 1(1):9–13, February 2003.
20. David W. Corne and Joshua D. Knowles. The Pareto-envelope based selection algorithm for multiobjective optimization. In *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature (PPSN VI)*, pages 839–848, Berlin, 2000. Springer-Verlag.
21. David W. Corne and Joshua D. Knowles. No Free Lunch and Free Leftovers Theorems for Multiobjective Optimisation Problems. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 327–341, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.
22. Dragan Cvetković and Ian C. Parmee. Use of Preferences for GA–based Multi–objective Optimisation. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, and Robert E. Smith, editors, *GECCO–99: Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1504–1509, Orlando, Florida, USA, 1999. Morgan Kaufmann Publishers.
23. P. Czyzak and A. Jaszkiewicz. Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7:34–47, 1998.
24. N. V. Dakev, A. J. Chipperfield, J. F. Whidborne, and P. J. Fleming. An evolutionary algorithm approach for solving optimal control problems. In *Proceedings of the 13th International Federation of Automatic Control (IFAC) World Congress*, San Francisco, California, 1996.
25. Kalyanmoy Deb. Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*, 7(3):205–230, Fall 1999.
26. Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T Meyarivan. Fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *Parallel Problem Solving from Nature - PPSN VI : 6th International Conference Proceedings*, number 1917 in LNCS, pages 849–858. Springer, 2000.
27. Kalyanmoy Deb and Tushar Goel. Controlled Elitist Non-dominated Sorting Genetic Algorithms for Better Convergence. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 67–81. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.
28. Kalyanmoy Deb and Sachin Jain. Running performance metrics for evolutionary multi-objective optimization. Technical report, KANGAL, IIT Kanpur, India, May 2002.
29. Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable Multi-Objective Optimization Test Problems. In *Congress on Evolutionary*

Computation (CEC'2002), volume 1, pages 825–830, Piscataway, New Jersey, May 2002. IEEE Service Center.

30. M. Dorigo and G. Di Caro. The Ant Colony Optimization meta-heuristic. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*. McGraw-Hill, 1999.
31. Nicole Drechsler, Rolf Drechsler, and Bernd Becker. Multi-objective Optimisation Based on Relation *favour*. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 154–166. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.
32. Matthias Ehrgott and Xavier Gandibleux. A Survey and Annotated Bibliography of Multiobjective Combinatorial Optimization. *OR Spektrum*, 22:425–460, 2000.
33. Ágoston E. Eiben, Robert Hinterding, and Zbigniew Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3:124–141, 1999.
34. M. Farina and P. Amato. Fuzzy Optimality and Evolutionary Multiobjective Optimization. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 58–72, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.
35. M. Farina, K. Deb, and P. Amato. Dynamic Multiobjective Optimization Problems: Test Cases, Approximation, and Applications. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 311–326, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.
36. Jonathan E. Fieldsend, Richard M. Everson, and Sameer Singh. Using Unconstrained Elite Archives for Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 7(3):305–323, June 2003.
37. M. Fleischer. The Measure of Pareto Optima. Applications to Multi-objective Metaheuristics. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 519–533, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.
38. Peter Fleming. Designing Control Systems with Multiple Objectives. In *IEE Colloquium on Advances in Control Technology*, pages 4/1–4/4, 1999.
39. Carlos M. Fonseca and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.
40. Carlos M. Fonseca and Peter J. Fleming. An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, 3(1):1–16, Spring 1995.
41. C.M. Fonseca and P.J. Fleming. Multiobjective optimal controller design with genetic algorithms. In *International Conference on Control*, volume 1, pages 745–749, 1994.

42. Luca Maria Gambardella, Éric Taillard, and Giovanni Agazzi. MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. In David Corne, Marco Dorigo, and Fred Glover, editors, *New Ideas in Optimization*, pages 63–76. McGraw-Hill, 1999.
43. Xavier Gandibleux, Hiroyuki Morita, and Naoki Katoh. The Supported Solutions Used as a Genetic Information in a Population Heuristic. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 429–442. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.
44. David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.
45. D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms and their Applications*, pages 41–49. Lawrence Erlbaum, 1987.
46. Marc Gravel, Wilson L. Price, and Caroline Gagné. Scheduling Continuous Casting of Aluminum Using a Multiple-Objective Ant Colony Optimization Metaheuristic. Technical Report 2001–004, Faculté des Sciences de L'Administration, Université Laval, Québec, Canada, April 2001. Available at `http://www.fsa.ulaval.ca/rd`.
47. Michael Guntsch and Martin Middendorf. Solving Multi-criteria Optimization Problems with Population-Based ACO. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 464–478, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.
48. D. Halhal, G. A. Walters, D. Ouazar, and D. A. Savic. Multi-objective improvement of water distribution systems using a structure messy genetic algorithm approach. *Journal of Water Resources Planning and Management ASCE*, 123(3):137–146, 1997.
49. Michael Pilegaard Hansen. Tabu Search in Multiobjective Optimisation : MOTS. In *Proceedings of the 13th International Conference on Multiple Criteria Decision Making (MCDM'97)*, Cape Town, South Africa, January 1997.
50. Michael Pilegaard Hansen and Andrzej Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Technical University of Denmark, March 1998.
51. M. Hapke, A. Jaszkiewicz, and R. Slowinski. Fuzzy multi-mode resource-constrained project scheduling with multiple objectives. In J. Weglarz, editor, *Recent Advances in Project Scheduling*, chapter 16, pages 355–382. Kluwer Academic Publishers, 1998.
52. Jeffrey Horn. Multicriterion Decision Making. In Thomas Bäck, David Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, volume 1, pages F1.9:1 – F1.9:15. IOP Publishing Ltd. and Oxford University Press, 1997.
53. Jeffrey Horn and Nicholas Nafpliotis. Multiobjective Optimization using the Niched Pareto Genetic Algorithm. Technical Report IlliGAl Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

54. Hisao Ishibuchi and Tadahiko Murata. Multi-Objective Genetic Local Search Algorithm. In Toshio Fukuda and Takeshi Furuhashi, editors, *Proceedings of the 1996 International Conference on Evolutionary Computation*, pages 119–124, Nagoya, Japan, 1996. IEEE.
55. Hisao Ishibuchi and Tadahiko Murata. Multi-Objective Genetic Local Search Algorithm and Its Application to Flowshop Scheduling. *IEEE Transactions on Systems, Man and Cybernetics*, 28(3):392–403, August 1998.
56. Hisao Ishibuchi and Youhei Shibata. An Empirical Study on the Effect of Mating Restriction on the Search Ability of EMO Algorithms. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 433–477, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.
57. Hisao Ishibuchi and Tadashi Yoshida. Hybid Evolutionary Multi-Objective Optimization Algorithms. In A. Abraham, J. Ruiz del Solar, and M. Köppen, editors, *Soft Computing Systems: Design, Management and Applications (Frontiers in Artificial Intelligence and Applications, Volume 87)*, pages 163–172. IOS Press, ISBN 1-58603-297-6, 2002.
58. Hisao Ishibuchi, Tadashi Yoshida, and Tadahiko Murata. Selection of Initial Solutions for Local Search in Multiobjective Genetic Local Search. In *Congress on Evolutionary Computation (CEC'2002)*, volume 1, pages 950–955, Piscataway, New Jersey, May 2002. IEEE Service Center.
59. Hisao Ishibuchi, Tadashi Yoshida, and Tadahiko Murata. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *IEEE Transactions on Evolutionary Computation*, 7(2):204–223, April 2003.
60. Andrzej Jaszkiewicz. Genetic local search for multiple objective combinatorial optimization. Technical Report RA-014/98, Institute of Computing Science, Poznan University of Technology, 1998.
61. Andrzej Jaszkiewicz. Do multiple-objective metaheuristics deliver on their promises? a computational experiment on the set-covering problem. *IEEE Transactions on Evolutionary Computation*, 7(2):133–143, April 2003.
62. Mikkel T. Jensen. *Robust and Flexible Scheduling with Evolutionary Computation.* PhD thesis, Department of Computer Science. University of Aarhus, Aarhus, Denmark, October 2001.
63. Mikkel T. Jensen. Guiding single-objective optimization using multi-objective methods. In *Applications of Evolutionary Computation*, volume 2611 of *LNCS*, pages 268–279. Springer, 2003.
64. Mikkel T. Jensen. Reducing the run-time complexity of multi-objective eas: The nsga-ii and other algorithms. *IEEE Transactions on Evolutionary Computation*, 7(5):502–515, 2003.
65. Nazan Khan, David E. Goldberg, and Martin Pelikan. Multi-Objective Bayesian Optimization Algorithm. In W.B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M.A. Potter, A.C. Schultz, J.F. Miller, E. Burke, and N. Jonoska, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2002)*, page 684, San Francisco, California, July 2002. Morgan Kaufmann Publishers.

66. Michael Kirley. MEA: A metapopulation evolutionary algorithm for multi-objective optimisation problems. In *Proceedings of the Congress on Evolutionary Computation 2001 (CEC'2001)*, volume 2, pages 949–956, Piscataway, New Jersey, May 2001. IEEE Service Center.
67. Joshua Knowles and David Corne. M-PAES: A Memetic Algorithm for Multiobjective Optimization. In *2000 Congress on Evolutionary Computation*, volume 1, pages 325–332, Piscataway, New Jersey, July 2000. IEEE Service Center.
68. Joshua Knowles and David Corne. On Metrics for Comparing Nondominated Sets. In *Congress on Evolutionary Computation (CEC'2002)*, volume 1, pages 711–716, Piscataway, New Jersey, May 2002. IEEE Service Center.
69. Joshua Knowles and David Corne. Towards Landscape Analyses to Inform the Design of Hybrid Local Search for the Multiobjective Quadratic Assignment Problem. In A. Abraham, J. Ruiz del Solar, and M. Köppen, editors, *Soft Computing Systems: Design, Management and Applications*, pages 271–279, Amsterdam, 2002. IOS Press. ISBN 1-58603-297-6.
70. Joshua Knowles and David Corne. Properties of an Adaptive Archiving Algorithm for Storing Nondominated Vectors. *IEEE Transactions on Evolutionary Computation*, 7(2):100–116, April 2003.
71. Joshua Knowles and David Corne. Bounded Pareto archiving: Theory and practice. In X. Gandibleux, M. Sevaux, K. Sörensen, and V. T'Kindt, editors, *Metaheuristics for Multiobjective Optimisation*, volume 535 of *Lecture Notes in Economics and Mathematical Systems*. Springer, January 2004. To appear.
72. Joshua D. Knowles. *Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization*. PhD thesis, University of Reading, UK, 2002.
73. Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.
74. Joshua D. Knowles, David W. Corne, and Mark Fleischer. Bounded archiving using the Lebesgue measure. In *Proceedings of the IEEE Congress on Evolutionary Computation*. IEEE Press, 2003. To appear.
75. Joshua D. Knowles, David W. Corne, and Mark Fleischer. Bounded archiving using the lebesgue measure. In *Proceedings of the IEEE Congress on Evolutionary Computation*. IEEE Press, 2003. (in press).
76. Joshua D. Knowles, Richard A. Watson, and David W. Corne. Reducing local optima in single-objective problems by multi-objectivization. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *Evolutionary Multi-Criterion Optimization : first international conference; proceedings / EMO 2001*, volume 1993 of *LNCS*, pages 269–283. Springer, 2001.
77. Natalio Krasnogor. *Studies on the Theory and Design Space of Memetic Algorithms*. PhD thesis, University of Nottingham, 2002.
78. Frank Kursawe. A Variant of Evolution Strategies for Vector Optimization. In H. P. Schwefel and R. Männer, editors, *Parallel Problem Solving from Nature. 1st Workshop, PPSN I*, volume 496 of *Lecture Notes in Computer Science Vol. 496*, pages 193–197, Berlin, Germany, October 1991. Springer-Verlag.
79. W. B. Langdon and P. C. Treleaven. Scheduling Maintenance of Electrical Power Transmission Networks Using Genetic Programming. In Kevin Warwick, Arthur Ekwue, and Raj Aggarwal, editors, *Artificial Intelligence Techniques in Power Systems*, chapter 10, pages 220–237. IEE, 1997.

80. Marco Laumanns and Jiri Ocenasek. Bayesian Optimization Algorithms for Multi-objective Optimization. In Juan Julián Merelo Guervós, Panagiotis Adamidis, Hans-Georg Beyer, José-Luis Fernández-Villaca nas, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature—PPSN VII*, pages 298–307, Granada, Spain, September 2002. Springer-Verlag. Lecture Notes in Computer Science No. 2439.
81. Marco Laumanns, Lothar Thiele, Kalyanmoy Deb, and Eckart Zitzler. Combining convergence and diversity in evolutionary multi-objective optimization. *Evolutionary Computation*, 10(3):263–282, Fall 2002.
82. Marco Laumanns, Lothar Thiele, Eckart Zitzler, and Kalyanmoy Deb. Archiving with Guaranteed Convergence and Diversity in Multi-Objective Optimization. In W.B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M.A. Potter, A.C. Schultz, J.F. Miller, E. Burke, and N. Jonoska, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2002)*, pages 439–447, San Francisco, California, July 2002. Morgan Kaufmann Publishers.
83. Z. Michalewicz and D. B. Fogel. *How to Solve It: Modern Heuristics*. Springer-Verlag, 2000.
84. Kaisa Miettinen. Some methods for nonlinear multi-objective optimization. In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, and D. Corne, editors, *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001)*, volume 1993 of *Lecture Notes in Computer Science*, pages 1–20, Berlin, 2001. Springer-Verlag.
85. Pablo Moscato. *New Ideas in Optimization*, chapter Memetic Algorithms: A Short Introduction, pages 219–234. McGraw Hill, 1999.
86. Tadahiko Murata and Hisao Ishibuchi. Constructing Multi-Objective Genetic Local Search Algorithms for Multi-Objective Flowshop Scheduling Problems. In *Proceedings of the 1998 Japan-USA Symposium on Flexible Automation*, pages 1353–1356, Ohtsu, Japan, July 1998.
87. Tadahiko Murata, Hisao Ishibuchi, and Mitsuo Gen. Cellular Genetic Local Search for Multi-Objective Optimization. In Darrell Whitley, David Goldberg, Erick Cantú-Paz, Lee Spector, Ian Parmee, and Hans-Georg Beyer, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2000)*, pages 307–314, San Francisco, California, 2000. Morgan Kaufmann.
88. Tadahiko Murata, Hisao Ishibuchi, and Mitsuo Gen. Specification of Genetic Search Directions in Cellular Multi-objective Genetic Algorithms. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 82–95. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.
89. Tadahiko Murata, Shiori Kaige, and Hisao Ishibuchi. Generalization of Dominance Relation-Based Replacement Rules for Memetic EMO Algorithms. In Erick Cantú-Paz et al., editor, *Genetic and Evolutionary Computation—GECCO 2003. Proceedings, Part I*, pages 1234–1245. Springer. Lecture Notes in Computer Science Vol. 2723, July 2003.
90. Luis Paquete and Thomas Stützle. A Two-Phase Local Search for the Biobjective Traveling Salesman Problem. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary*

Multi-Criterion Optimization. Second International Conference, EMO 2003, pages 479–493, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.

91. Luís F. Paquete and Carlos M. Fonseca. A Study of Examination Timetabling with Multiobjective Evolutionary Algorithms. In Jorge Pinho de Sousa, editor, *Proceedings of the 4th Metaheuristics International Conference (MIC'2001)*, pages 149–153. Program Operational Ciencia, Tecnologia, Inovaçao do Quadro Comunitário de Apoio III de Fundaçao para a Ciencia e Tecnologia, Porto, Portugal, July 16–20 2001.
92. Kwang-Wook Park and Donald E. Grierson. Pareto-Optimal Conceptual Design of the Structural Layout of Buildings Using a Multicriteria Genetic Algorithm. *Computer-Aided Civil and Infrastructure Engineering*, 14(3):163–170, May 1999.
93. Sangbong Park, Dongkyung Nam, and Cheol Hoon Park. Design of a neural controller using multiobjective optimization for nonminimum phase systems. In *1999 IEEE International Fuzzy Systems Conference Proceedings*, volume 1, pages 533–537. IEEE, 1999.
94. Geoffrey T. Parks and I. Miller. Selective Breeding in a Multiobjective Genetic Algorithm. In A. E. Eiben, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving From Nature — PPSN V*, pages 250–259, Amsterdam, Holland, 1998. Springer-Verlag.
95. Ian C. Parmee, Dragan Cvetković, Andrew H. Watson, and Christopher R. Bonham. Multiobjective Satisfaction within an Interactive Evolutionary Design Environment. *Evolutionary Computation*, 8(2):197–222, Summer 2000.
96. Günter Rudolph. Evolutionary Search for Minimal Elements in Partially Ordered Finite Sets. In V.W. Porto, N. Saravanan, D. Waagen, and A.E. Eiben, editors, *Evolutionary Programming VII, Proceedings of the 7th Annual Conference on Evolutionary Programming*, pages 345–353, Berlin, 1998. Springer.
97. J. David Schaffer. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms.* PhD thesis, Vanderbilt University, 1984.
98. Josef Schwarz and Jiri Ocenasek. Evolutionary Multiobjective Bayesian Optimization Algorithm: Experimental Study. In *Proceedings of the 35th Spring International Conference: Modelling and Simulation of Systems (MOSIS'01)*, pages 101–108, Czech Republic, 2001. MARQ, Hradec and Moravici.
99. Paolo Serafini. Simulated Annealing for Multiple Objective Optimization Problems. In G.H. Tzeng, H.F. Wang, U.P. Wen, and P.L. Yu, editors, *Proceedings of the Tenth International Conference on Multiple Criteria Decision Making: Expand and Enrich the Domains of Thinking and Application*, volume 1, pages 283–294, Berlin, 1994. Springer-Verlag.
100. N. Srinivas and Kalyanmoy Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. Technical report, Department of Mechanical Engineering, Indian Institute of Technology, Kanpur, India, 1993.
101. Rainer Storn and Kenneth Price. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, International Computer Science Institute, Berkeley, CA, 1995.
102. Ricardo Szmit and Amnon Barak. Evolution Strategies for a Parallel Multi-Objective Genetic Algorithm. In Darrell Whitley, David Goldberg, Erick Cantú-Paz, Lee Spector, Ian Parmee, and Hans-Georg Beyer, editors, *Proceed-*

ings of the Genetic and Evolutionary Computation Conference (GECCO'2000), pages 227–234, San Francisco, California, 2000. Morgan Kaufmann.

103. El-Ghazali Talbi, Malek Rahoual, Mohamed Hakim Mabed, and Clarisse Dhaenens. A Hybrid Evolutionary Approach for Multicriteria Optimization Problems: Application to the Flow Shop. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 416–428. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.
104. Kay Chen Tan and Yun Li. Multi-Objective Genetic Algorithm Based Time and Frequency Domain Design Unification of Linear Control Systems. In *Proceedings of the IFAC/IEEE International Symposium on Artificial Intelligence in Real-Time Control*, pages 61–66, Kuala Lumpur, Malaysia, September 1997.
105. K.C. Tan, T.H. Lee, and E.F. Khor. Evolutionary Algorithms with Dynamic Population Size and Local Exploration for Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 5(6):565–588, December 2001.
106. Dirk Thierens and Peter A.N. Bosman. Multi-Objective Mixture-based Iterated Density Estimation Evolutionary Algorithms. In Lee Spector, Erik D. Goodman, Annie Wu, W.B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, pages 663–670, San Francisco, California, 2001. Morgan Kaufmann Publishers.
107. E.L. Ulungu, J. Teghem, Ph. Fortemps, and D. Tuyttens. MOSA Method: A Tool for Solving Multiobjective Combinatorial Optimization Problems. *Journal of Multi-Criteria Decision Analysis*, 8(4):221–236, 1999.
108. David A. Van Veldhuizen and Gary B. Lamont. Multiobjective Optimization with Messy Genetic Algorithms. In *Proceedings of the 2000 ACM Symposium on Applied Computing*, pages 470–476, Villa Olmo, Como, Italy, 2000. ACM.
109. David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1:67–82, 1997.
110. Kazuo Yamasaki. Dynamic Pareto Optimum GA against the changing environments. In *2001 Genetic and Evolutionary Computation Conference. Workshop Program*, pages 47–50, San Francisco, California, July 2001.
111. Gary G. Yen and Haiming Lu. Hierarchical Rank Density Genetic Algorithm for Radial-Basis Function Neural Network Design. In *Congress on Evolutionary Computation (CEC'2002)*, volume 1, pages 25–30, Piscataway, New Jersey, May 2002. IEEE Service Center.
112. Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications.* PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.
113. Eckart Zitzler and Lothar Thiele. An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, May 1998.
114. Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert da Fonseca. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, April 2003.
115. Jesse B. Zydallis, David A. Van Veldhuizen, and Gary B. Lamont. A Statistical Comparison of Multiobjective Evolutionary Algorithms Including

the MOMGA–II. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 226–240. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.