Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms I: A Unified Formulation

Carlos M. Fonseca[1] and Peter J. Fleming[2]
Dept. Automatic Control and Systems Eng.
University of Sheffield
Sheffield S1 4DU, U.K.

January 23, 1995

Research Report 564

[1]C.Fonseca@shef.ac.uk
[2]P.Fleming@shef.ac.uk

Contents

Abstract

In optimization, multiple objectives and constraints cannot be handled independently of the underlying optimizer. Requirements such as continuity and differentiability of the cost surface add yet another conflicting element to the decision process. While "better" solutions should be rated higher than "worse" ones, the resulting cost landscape must also comply with such requirements.

Evolutionary algorithms (EAs), which have found application in many areas not amenable to optimization by other methods, possess many characteristics desirable in a multiobjective optimizer, most notably the concerted handling of multiple candidate solutions. However, EAs are essentially unconstrained search techniques which require the assignment of a scalar measure of quality, or fitness, to such candidate solutions.

After reviewing current evolutionary approaches to multiobjective and constrained optimization, the paper proposes that fitness assignment be interpreted as, or at least related to, a multicriterion decision process. A suitable decision making framework based on goals and priorities is subsequently formulated in terms of a relational operator, characterized, and shown to encompass a number of simpler decision strategies. Finally, the ranking of an arbitrary number of candidates is considered. The effect of preference changes on the cost surface seen by an EA is illustrated graphically for a simple problem.

The paper concludes with the formulation of a multiobjective genetic algorithm based on the proposed decision strategy. Niche formation techniques are used to promote diversity among preferable candidates, and progressive articulation of preferences is shown to be possible as long as the genetic algorithm can recover from abrupt changes in the cost landscape.

1 Introduction

Constraint satisfaction and multiobjective optimization are very much two aspects of the same problem. Both involve the simultaneous optimization of a number of functions. Constraints can often be seen as hard objectives, which need to be satisfied before the optimization of the remaining, soft, objectives takes place. Conversely, problems characterized by a number of soft objectives are often re-formulated as constrained optimization problems in order to be solved.

Despite having been successfully used to approach many ill-behaved problems, the first formulations of evolutionary algorithms were essentially single-function methods with little scope for constraint handling. Following the success of the evolutionary approach, interest in how both constraints and multiple objectives can be handled by evolutionary algorithms has rapidly increased.

Multiobjective and constrained optimization are introduced here separately, first in general terms, and then in the context of evolutionary algorithms. Current practices are then presented and discussed.

The formulation and characterization of a unified decision making framework for multi-function optimization follows, encompassing both objectives and constraints. Finally, a Multiobjective Genetic Algorithm is described, and presented as a method which can be used for progressive articulation of preferences.

2 Constrained optimization

Practical problems often see their solution constrained by a number of restrictions imposed on the decision variables. Constraints usually fall into one of two different categories:

Domain constraints express the domain of definition of the objective function. In control systems, closed-loop system stability is an example of a domain constraint, because most performance measures are not defined for unstable systems.

Preference constraints impose further restrictions on the solution of the problem according to knowledge at a higher level. A given stability margin, for example, expresses a preference of the designer.

Constraints can usually be expressed in terms of function inequalities of the type

$$f(\mathbf{x}) \leq g$$

where f is a, generally non-linear, real-valued function of the decision variable vector $\mathbf{x}$ and g is a constant value. The inequality may also be strict ($<$ instead of $\leq$). Equality constraints of the type

$$f(\mathbf{x}) = g$$

can be formulated as particular cases of inequality constraints.

Without loss of generality, the constrained optimization problem is that of minimizing a scalar function f_1 of some decision variable vector $\mathbf{x}$ in a universe $\mathcal{U}$, subject to a number $n-1$ of conditions involving $\mathbf{x}$, and eventually expressed as a functional vector inequality of the type

$$(f_2(\mathbf{x}), \ldots, f_n(\mathbf{x})) \leq (g_2, \ldots, g_n)$$

where the inequality applies component-by-component. It is assumed that there is at least one point in $\mathcal{U}$ which satisfies all constraints.

In many cases, satisfying constraints is a difficult problem in itself. When constraints cannot be all simultaneously satisfied, the problem is often deemed to admit no solution. The number of constraints violated, and the extent to which each constraint is violated, then needs to be considered in order to relax the preference constraints.

3 Multiobjective optimization

Many problems are also characterized by several non-commensurable and often competing measures of performance, or objectives. The multiobjective optimization problem is, without loss of generality, the problem of simultaneously minimizing the n components f_k, $k = 1, \ldots, n$, of a vector function $\mathbf{f}$ of a variable $\mathbf{x}$ in a universe $\mathcal{U}$, where

$$\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_n(\mathbf{x})).$$

The problem has usually no unique, perfect solution, but a set of equally efficient, or non-inferior, alternative solutions, known as the Pareto-optimal set[1]. Still assuming a minimization problem, inferiority is defined as follows:

Definition 1 (inferiority) *A vector* $\mathbf{u} = (u_1, \ldots, u_n)$ *is said to be inferior to* $\mathbf{v} = (v_1, \ldots, v_n)$ *iff* $\mathbf{v}$ *is partially less than* $\mathbf{u}$ *(*$\mathbf{v} \;_{\mathrm{P}}\!< \mathbf{u}$*), i.e.,*

$$\forall\, i \in \{1, \ldots, n\}\,,\ v_i \leq u_i \quad \wedge \quad \exists\, i \in \{1, \ldots, n\} \mid v_i < u_i$$

Alternatively, $\mathbf{v}$ *can be said to be superior to, or to dominate,* $\mathbf{u}$.

Definition 2 (non-inferiority) *Vectors* $\mathbf{u} = (u_1, \ldots, u_n)$ *and* $\mathbf{v} = (v_1, \ldots, v_n)$ *are said to be non-inferior to each other if neither* $\mathbf{v}$ *is inferior to* $\mathbf{u}$ *nor* $\mathbf{u}$ *is inferior to* $\mathbf{v}$.

The notion of non-inferiority is only a first step towards solving an MO problem. In order to select a suitable compromise solution from all non-inferior alternatives, a decision process is also necessary.

Depending on how the computation and the decision processes are combined in the search for compromise solutions, three broad classes of MO methods exist [2]:

A priori articulation of preferences The decision maker expresses preferences in terms of an aggregating function which combines individual objective values into a single utility value, and ultimately makes the problem single-objective, prior to optimization.

A posteriori articulation of preferences The decision maker is presented by the optimizer with a set of candidate non-inferior solutions, before expressing any preferences. The compromise solution is chosen from that set.

Progressive articulation of preferences Decision making and optimization occur at interleaved steps. At each step, partial preference information is supplied by the decision maker to the optimizer, which, in turn, generates better alternatives according to the information received.

3.1 Preference articulation

Independently of the stage at which it takes place, preference articulation implicitly defines a so-called *utility function* which discriminates between candidate solutions. Although such a utility function can be very difficult to formalize in every detail, approaches based on the following have been widely used.

Weighting coefficients are real values which express the relative importance of the objectives and control their involvement in the overall utility measure. The weighted-sum approach is the classical example of a method based on objective weighting [2].

Priorities are integer values which determine in which order objectives are to be optimized, according to their importance. The lexicographic method [1], for example, requires all objectives to be assigned different priorities.

Goal values indicate desired levels of performance in each objective dimension. The way in which goals are interpreted may vary. In particular, they may represent minimum levels of performance to be attained, utopian performance levels to be approximated, or ideal performance levels to be matched as closely as possible [3]. Goals are usually easier to set than weights and priorities, because they relate more closely to the final solution of the problem.

3.2 Constraint satisfaction as a multiobjective problem

The problem of satisfying a number of violated inequality constraints is clearly the multiobjective problem of minimizing the associated functions until given values (goals) are reached. The concept of non-inferiority is readily applicable and particularly appropriate when constraints are themselves non-commensurable. When not all goals can be simultaneously met, a *family* of violating, non-inferior points is the closest to a solution of the problem.

Goal-based multiobjective optimization extends simple constraint satisfaction in the sense that the optimization continues even after all goals are met. In this case, solutions should both be non-inferior and meet all goals.

4 Overview of evolutionary approaches to multi-function optimization

The term Evolutionary Algorithms (EAs) is used to refer to a number of search and optimization algorithms inspired by the process of natural evolution. Current evolutionary approaches include Evolutionary Programming (EP) [4], Evolution Strategies (ESs) [5], Genetic Algorithms (GAs) [6] and Genetic Programming (GP) [7]. A comparative study of the first three approaches can be found in [8].

Evolutionary algorithms maintain a population of candidate solutions (the individuals) for a given problem. Individuals are evaluated and assigned fitness values based on their relative performance. They are then given a chance to reproduce, i.e. replicate themselves a number of times proportional to their fitness. The offspring produced are modified by means of mutation and/or recombination operators before they are evaluated, and subsequently re-inserted in the population. Several re-insertion strategies exist, ranging from the unconditional replacement of the parents by the offspring to approaches where offspring replace the worst parents, their own parents or even the oldest parents.

The multiple performance measures provided by constrained and multiobjective problems must be converted into a scalar fitness measure before EAs can be applied. So far, constrained optimization has been considered separately from multiobjective objective optimization in EA literature, and, for that reason, the two are reviewed separately here.

4.1 Constraint handling

The simplest approach to handling constraints in EAs has been to assign infeasible individuals an arbitrarily low fitness [6, p. 85]. This is possible given the ability of EAs to cope with discontinuities, which arise on the constraint boundaries. In this approach, provided feasible solutions can be easily found, any infeasible individuals are selected out and the search is not affected much.

Certain types of constraints, however, such as bounds on the decision variables and other linear constraints, can be handled by mapping the search space so as to minimize the number of infeasible solutions it contains and/or designing the mutation and recombination operators carefully in order to minimize the production of infeasible offspring from feasible parents [9]. This and the previous approach are complementary and often used in combination with each other.

In the case where no feasible individuals are known, and cannot easily be found, simply assigning low-fitness to infeasible individuals makes the initial stages of evolution degenerate into a random walk. To avoid this, the penalty imposed onto infeasible individuals can be made to depend on the extent to which they violate the constraints. Such penalty values are typically added to the (unconstrained) performance value before fitness is computed [6, p. 85f].

Although penalty functions do provide a way of guiding the search towards feasible solutions when these are not known, they are very much problem dependent. Some infeasible solutions can, despite the penalty, be seen as better than some feasible ones, which can make the population evolve towards a false optimum. In response to these difficulties, guidelines on the use of penalty functions have been described by Richardson et al. [10].

One of the most recent approaches to constraint handling has been proposed by Powell and Skolnick [11] and consists of rescaling the original objective function to assume values less than unity in the feasible region, whilst assigning infeasible individuals penalty values greater than one. Subsequent ranking of the population correctly assigns higher fitness to all feasible points than to those infeasible. This perspective is supported and extended in the present work.

4.2 Multiple objectives

In problems where no global criterion directly emerges from the original multiobjective formulation, objectives are often artificially combined by means of an aggregating function. Many such approaches, although initially developed to be used with other optimizers, can also be used with EAs.

Optimizing a combination of the objectives has the advantage of producing a single compromise solution, requiring no further interaction with the decision maker. However, if the solution found cannot be accepted as a good compromise, tuning of the aggregating function may be required, followed by new runs of the optimizer, until a suitable solution is found. As a workaround, of the many candidate solutions evaluated in a single run of the EA, those non-dominated solutions may provide valuable alternatives [12, 13]. However, since the algorithm sees such alternatives as sub-optimal, they cannot be expected to be optimal in any sense.

Aggregating functions have been widely used with EAs, from the simple weighted sum approach, e.g., [14], to target vector optimization [15]. An implementation of goal attainment, among other methods, was used by Wilson and Macleod [12].

4.2.1 Non-Pareto approaches

Treating objectives separately was first proposed by Schaffer [16], as a move towards finding multiple non-dominated solutions with a single algorithm run. In his approach, known as the Vector Evaluated Genetic Algorithm (VEGA), appropriate fractions of the next generation, or sub-populations, were selected according to each of the objectives, separately. Crossover and mutation were applied as usual after shuffling all the sub-populations together. Non-dominated individuals were identified by monitoring the population as it evolved.

In a more application oriented paper, Fourman [17] also chose not to combine the different objectives. Selection was performed by comparing pairs of individuals, each pair according to one objective selected at random. Fourman first experimented with assigning different priorities to the objectives and comparing individuals lexically, but found selecting objectives randomly to work "surprisingly" well.

However, shuffling sub-populations together, or having different objectives affecting different tournaments, corresponds to averaging the fitness components associated with each of the objectives. Since Schaffer used proportional fitness assignment, the resulting expected fitness corresponded, in fact, to a linear combination of the objectives with variable weights, as noted in [10]. Fourman's approach, on the other hand, corresponds to an averaging of rank, not objective, values. Different non-dominated individuals are, in both cases, generally assigned different fitness values, but the performance of the algorithms on problems with concave trade-off surfaces can be qualitatively different [18].

Another approach to selection based on the use of single objectives in alternation has been proposed in the context of ESs by Kursawe [19]. Hajela and Lin [20] elaborated on the VEGA by explicitly including sets of weights in the chromosome.

4.2.2 Pareto-based approaches

Another class of approaches, based on ranking according to the actual concept of Pareto optimality, was proposed later by Goldberg [6, p. 201], guaranteeing equal probability of reproduction to all non-dominated individuals. Problems with non-convex trade-off surfaces, which present difficulties to pure weighted-sum approaches, do not raise any special issues in Pareto optimization.

This paper elaborates on Pareto-based ranking by combining dominance with preference information to produce a suitable fitness assignment strategy. The evolutionary optimization process is seen as the result of the interaction between an artificial selector, here referred to as the Decision Maker (DM), and an evolutionary search process. The search process generates a new set of candidate solutions according to the utility assigned by the DM to the current set of candidates.

Whilst the action of the DM influences the production of new individuals, these, as they are evaluated, provide new trade-off information which the DM can use to refine its current preferences. The EA sees the effect of any changes in the decision process, which may or may not result from taking recently acquired information into account, as an environmental change. This general view of multiobjective evolutionary optimization has been proposed by the authors in earlier work [21] and is illustrated in Figure 1. The DM block represents any utility assignment strategy, which may range from an intelligent decision maker to a simple weighted sum approach.

The EA block is concerned with a different, but complementary, aspect of the optimization, the search process. Evolutionary algorithms, in the first instance, make very few assumptions about the fitness landscape they work on, which jus-

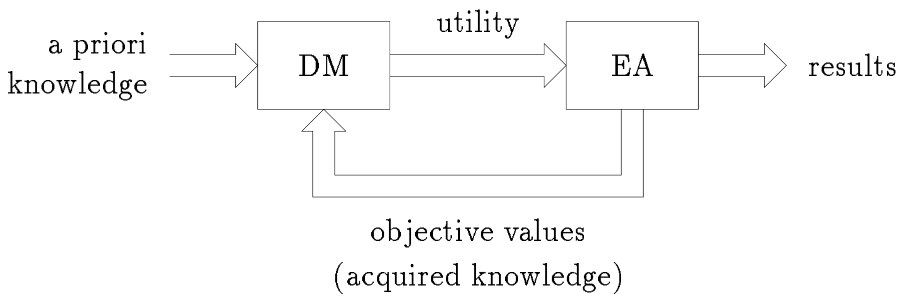


Figure 1: A general multiobjective evolutionary optimizer

tifies and permits a primary concern with fitness assignment. However, EAs are not capable of optimizing arbitrary functions [22]. Some form of characterization of the multiobjective fitness landscapes associated with the decision making strategy used is, therefore, important, and the design of the EA should take that information into account.

5 Multiobjective decision making based on given goals and priorities

The specification of goals and priorities can accommodate a whole variety of constrained and/or multiobjective problem formulations. Goal and priority information is often naturally available from the problem formulation, although not necessarily in a strict sense. Therefore, the interpretation of such information should take its partial character into account. This can be accomplished by allowing different objectives to be given the same priority, and by avoiding using measures of the distance to the goals, which inevitably depend on the scale in which the objective values are presented.

An extension of the decision making strategy proposed by the authors in [21] is formulated here in terms of a relational operator, which incorporates the preference information given, and characterized. The ranking of a whole population based on such a relation is then described.

5.1 The comparison operator

Consider an n-dimensional vector function $\mathbf{f}$ of some decision variable $\mathbf{x}$ and two n-dimensional objective vectors $\mathbf{u} = \mathbf{f}(\mathbf{x_u})$ and $\mathbf{v} = \mathbf{f}(\mathbf{x_v})$, where $\mathbf{x_u}$ and $\mathbf{x_v}$ are particular values of $\mathbf{x}$. Consider also the n-dimensional *preference vector*

$$\begin{aligned} \mathbf{g} &= [\mathbf{g}_1, \ldots, \mathbf{g}_p] \\ &= [(g_{1,1}, \ldots, g_{1,n_1}), \ldots, (g_{p,1}, \ldots, g_{p,n_p})] \end{aligned}$$

where $n_i \in \{0, \ldots, n\}$ for $i = 1, \ldots, p$, and

$$\sum_{i=1}^{p} n_i = n\,.$$

Similarly, $\mathbf{u}$ may be written as

$$\begin{aligned} \mathbf{u} &= [\mathbf{u}_1, \ldots, \mathbf{u}_p] \\ &= [(u_{1,1}, \ldots, u_{1,n_1}), \ldots, (u_{p,1}, \ldots, u_{p,n_p})]\,, \end{aligned}$$

and the same for $\mathbf{v}$ and $\mathbf{f}$.

The sub-vectors $\mathbf{g}_i$ of the preference vector $\mathbf{g}$, where $i = 1, \ldots, p$, associate priorities i and goals g_{i,j_i}, where $j_i = 1, \ldots, n_i$, to the corresponding objective functions f_{i,j_i}, components of $\mathbf{f}_i$. This assumes a convenient permutation of the components of $\mathbf{f}$, without loss of generality.

Generally, each sub-vector $\mathbf{u}_i$ will be such that a number $k_i \in \{0, \ldots, n_i\}$ of its components meet their goals while the remaining do not. Also without loss of generality, $\mathbf{u}$ is such that, for $i = 1, \ldots, p$, one can write

$$\exists\, k_i \in \{0, \ldots, n_i\} \mid \forall \ell \in \{1, \ldots, k_i\}\,, \quad \forall m \in \{k_i + 1, \ldots, n_i\}\,, \qquad (u_{i,\ell} \le g_{i,\ell}) \wedge (u_{i,m} > g_{i,m}) \quad (1)$$

For simplicity, the first k_i components of vectors $\mathbf{u}_i$, $\mathbf{v}_i$ and $\mathbf{g}_i$ will be represented as $\mathbf{u}_i^{\overset{\mathbf{u}}{\smile}}$, $\mathbf{v}_i^{\overset{\mathbf{u}}{\smile}}$ and $\mathbf{g}_i^{\overset{\mathbf{u}}{\smile}}$, respectively. The last $n_i - k_i$ components of the same vectors will be denoted $\mathbf{u}_i^{\overset{\mathbf{u}}{\frown}}$, $\mathbf{v}_i^{\overset{\mathbf{u}}{\frown}}$ and $\mathbf{g}_i^{\overset{\mathbf{u}}{\frown}}$, also respectively. The smile ($\overset{\mathbf{u}}{\smile}$) and the frown ($\overset{\mathbf{u}}{\frown}$), respectively, indicate the components in which $\mathbf{u}$ either does or does not meet the goals.

Definition 3 (preferability) *Vector* $\mathbf{u} = [\mathbf{u}_1, \ldots, \mathbf{u}_p]$ *is preferable to* $\mathbf{v} = [\mathbf{v}_1, \ldots, \mathbf{v}_p]$ *given a preference vector* $\mathbf{g} = [\mathbf{g}_1, \ldots, \mathbf{g}_p]$ *(*$\mathbf{u} \underset{\mathbf{g}}{\prec} \mathbf{v}$*) iff*

$$p = 1 \;\Rightarrow\; (\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \mathrel{{}_{\mathrm{p}}{<}} \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \vee \left\{ (\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \wedge \left[(\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}}) \vee (\mathbf{u}_p^{\overset{\mathbf{u}}{\smile}} \mathrel{{}_{\mathrm{p}}{<}} \mathbf{v}_p^{\overset{\mathbf{u}}{\smile}}) \right] \right\}$$

and

$$p > 1 \;\Rightarrow\; (\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \mathrel{{}_{\mathrm{p}}{<}} \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \vee \left\{ (\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \wedge \left[(\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}}) \vee (\mathbf{u}_{1,\ldots,p-1} \underset{\mathbf{g}_{1,\ldots,p-1}}{\prec} \mathbf{v}_{1,\ldots,p-1}) \right] \right\}$$

where $\mathbf{u}_{1,\ldots,p-1} = [\mathbf{u}_1, \ldots, \mathbf{u}_{p-1}]$ *and similarly for* $\mathbf{v}$ *and* $\mathbf{g}$*.*

In simple terms, vectors $\mathbf{u}$ and $\mathbf{v}$ are compared first in terms of their components with the highest priority, that is, those where $i = p$, disregarding those in which $\mathbf{u}_p$ meets the corresponding goals, $\mathbf{u}_p^{\overset{\mathbf{u}}{\smile}}$. In case both vectors meet all goals with this priority, or if they violate some or all of them, but in exactly the same way, the next priority level $(p-1)$ is considered. The process continues until priority 1 is reached and satisfied, in which case the result is decided by comparing the priority 1 components of the two vectors in a Pareto fashion.

Since satisfied high-priority objectives are left out from comparison, vectors which are equal to each other in all but these components express virtually no trade-off information given the corresponding preferences. The following symmetric relation is defined:

Definition 4 (equivalence) *Vector* $\mathbf{u} = [\mathbf{u}_1, \ldots, \mathbf{u}_p]$ *is equivalent to* $\mathbf{v} = [\mathbf{v}_1, \ldots, \mathbf{v}_p]$ *given a preference vector* $\mathbf{g} = [\mathbf{g}_1, \ldots, \mathbf{g}_p]$ *(*$\mathbf{u} \underset{\mathbf{g}}{\equiv} \mathbf{v}$*) iff*

$$(\mathbf{u}^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}^{\overset{\mathbf{u}}{\frown}}) \wedge (\mathbf{u}_1^{\overset{\mathbf{u}}{\smile}} = \mathbf{v}_1^{\overset{\mathbf{u}}{\smile}}) \wedge (\mathbf{v}_{2,\ldots,p}^{\overset{\mathbf{u}}{\smile}} \leq \mathbf{g}_{2,\ldots,p}^{\overset{\mathbf{u}}{\smile}}) \,.$$

The concept of preferability can be related to that of inferiority as follows:

Lemma 1 *For any two objective vectors* $\mathbf{u}$ *and* $\mathbf{v}$*, if* $\mathbf{u}\ {}_{\mathrm{P}}\!<\ \mathbf{v}$*, then* $\mathbf{u}$ *is either* preferable *or* equivalent *to* $\mathbf{v}$*, given any preference vector* $\mathbf{g} = [\mathbf{g}_1, \ldots, \mathbf{g}_p]$*.*

The proof of this lemma, and that of the following one, can be found in the Appendix.

Lemma 2 (transitivity) *The preferability relation is transitive, i.e., given any three objective vectors* $\mathbf{u}$*,* $\mathbf{v}$ *and* $\mathbf{w}$*, and a preference vector* $\mathbf{g} = [\mathbf{g}_1, \ldots, \mathbf{g}_p]$*,*

$$\mathbf{u} \underset{\mathbf{g}}{\prec} \mathbf{v} \underset{\mathbf{g}}{\prec} \mathbf{w} \quad \underset{\mathbf{g}}{\Longrightarrow} \quad \mathbf{u} \underset{\mathbf{g}}{\prec} \mathbf{w}\,.$$

5.1.1 Particular cases

The decision strategy described above encompasses a number of simpler multi-objective decision strategies, which correspond to particular settings of the preference vector.

Pareto (Definition 2) All objectives have equal priority and no goal levels are given. $\mathbf{g} = [\mathbf{g}_1] = [(-\infty, \ldots, -\infty)]$.

Lexicographic [1] Objectives are all assigned different priorities and no goal levels are given. $\mathbf{g} = [\mathbf{g}_1, \ldots, \mathbf{g}_n] = [(-\infty), \ldots, (-\infty)]$.

Constrained optimization (Section 2) The functional parts of a number n_c of inequality constraints are handled as high priority objectives to be minimized until the corresponding constant parts, the goals, are reached. The objective function is assigned the lowest priority. $\mathbf{g} = [\mathbf{g}_1, \mathbf{g}_2] = [(-\infty), (g_{2,1}, \ldots, g_{2,n_c})]$.

Constraint satisfaction (or Method of Inequalities [23]) All constraints are treated as in constrained optimization, but there is no low priority objective to be optimized. $\mathbf{g} = [\mathbf{g}_2] = [(g_{2,1}, \ldots, g_{2,n})]$.

Goal programming Several interpretations of goal programming can be implemented. A simple formulation, described in [2], consists of attempting to meet the goals sequentially, in a similar way to lexicographic optimization. $\mathbf{g} = [\mathbf{g}_1, \ldots, \mathbf{g}_n] = [(g_{1,1}), \ldots, (g_{n,1})]$.

A second formulation attempts to meet all the goals simultaneously, as with constraint satisfaction, but requires solutions to be satisfactory and Pareto optimal. $\mathbf{g} = [\mathbf{g}_1] = [(g_{1,1}, \ldots, g_{1,n})]$.

Aggregating functions, such as weighted sums and the maximum of a number of objectives, can, of course, be used as individual objectives. Although this may be appropriate in the case where they express some global criterion, e.g., financial cost, they do have the disadvantage of hiding information from the decision maker. It is especially worth pointing out that, as the number of objectives increases, it becomes more likely that some objectives are, in fact, non-competing, at least in portions of the trade-off surface. The understanding that some objectives are non-competing constitutes a valuable insight into the problem, because the number of dimensions involved in the trade-off is reduced.

5.2 Population ranking

As opposed to the single objective case, the ranking of a population in the multiobjective case is not unique. This is due to concepts such as dominance and preferability not defining total, but partial orders. In the present case, it is desired that all preferred individuals be assigned the same rank, and that individuals be placed higher in the rank than those they are preferable to.

Consider an individual $\mathbf{x_u}$ at generation t and with corresponding objective vector $\mathbf{u}$, and let $r_{\mathbf{u}}^{(t)}$ be the number of individuals in the current population which are preferable to it. The current position of $\mathbf{x_u}$ in the individuals' rank can be given simply by

$$\operatorname{rank}(\mathbf{x_u}, t) = r_{\mathbf{u}}^{(t)}$$

which ensures that all preferred individuals in the current population are assigned rank zero.

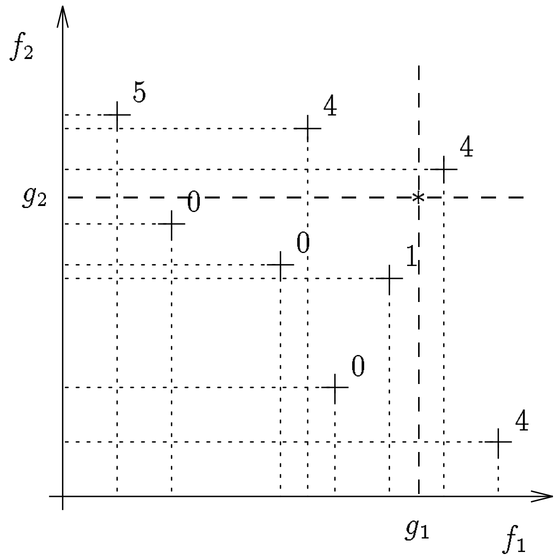
In the case of a large and uniformly distributed population with N individuals, the normalized rank $r^{(t)}/N$ constitutes an estimate of the fraction of the search space preferable to each individual considered. Such a fraction indicates how easily the current solution can be improved by pure random search and, as a measure of individual cost, it does not depend on how the objectives are scaled. This interpretation of ranking, also valid when there is only one objective, provides a way of characterizing the cost landscape associated with the preferences of the DM. It is not applicable to the ranking approach proposed by Goldberg [6, p. 201].

In the general case of a non-uniformly distributed population, a biased estimate is obtained which, nevertheless, preserves the strict order relationships between individuals, as desired.

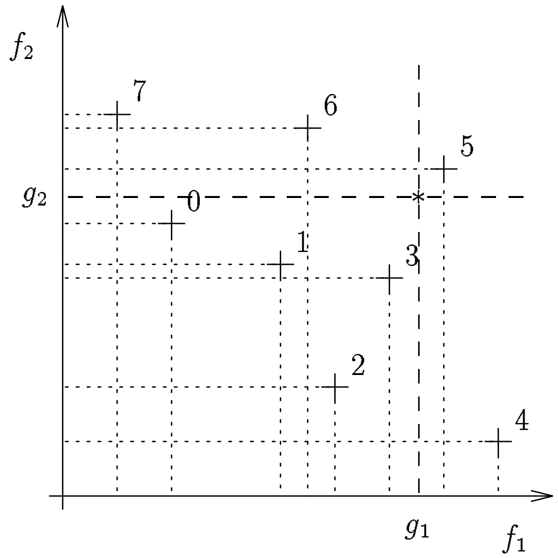
Lemma 3 *If an objective vector* $\mathbf{u} = \mathbf{f}(\mathrm{x}_{\mathbf{u}})$ *associated with an individual* $\mathrm{x}_{\mathbf{u}}$ *is preferable to another vector* $\mathbf{v} = \mathbf{f}(\mathrm{x}_{\mathbf{v}})$ *associated with an individual* $\mathrm{x}_{\mathbf{v}}$ *in the same arbitrary population, then* $\mathrm{rank}(\mathrm{x}_{\mathbf{u}}, t) < \mathrm{rank}(\mathrm{x}_{\mathbf{v}}, t)$. *Equivalently, if* $\mathrm{rank}(\mathrm{x}_{\mathbf{u}}, t) \geq \mathrm{rank}(\mathrm{x}_{\mathbf{v}}, t)$, *then* $\mathbf{u}$ *is not preferable to* $\mathbf{v}$.

The proof follows from the transitivity of the preferability relation (Lemma 2).

Figure 2 illustrates the ranking of the same population for two different preference vectors. In the first case, both objectives are given the same priority. Note that all satisficing individuals (the ones which meet their goals) are preferable to, and therefore have lower rank than, all of the remaining ones. In the second case, objective 2 is given a higher priority, reflecting, for example, a feasibility constraint. In this case, individuals which do not meet goal g_2 are the worst (they are infeasible), independently of their "theoretical" performance according to f_1. Once g_2 is met, f_1 is used for ranking. Individuals which meet both goals are satisficing solutions, whereas those which meet only g_2 are feasible, but unsatis-



(a) f_2 has the same priority as f_1



(b) f_2 has greater priority than f_1

Figure 2: Multiobjective ranking with goal values (minimization).

factory. Note how particular ranks need not be represented in the population at each particular generation.

5.3 Characterization of multiobjective cost landscapes

The cost landscape associated with a problem involving multiple objectives depends not only on the objectives themselves, but also on the preferences expressed by the DM. Their effect can be more easily understood by means of an example. Consider the simple bi-objective problem of simultaneously minimizing

$$\begin{aligned} f_1(x_1, x_2) &= 1 - \exp\left(-(x_1 - 1)^2 - (x_2 + 1)^2\right) \\ f_2(x_1, x_2) &= 1 - \exp\left(-(x_1 + 1)^2 - (x_2 - 1)^2\right) \end{aligned}$$

As suggested in the previous subsection, the cost landscape associated with a given set of preferences can be inferred from the ranking of a large, uniformly distributed population, and since the problem involves only two decision variables, visualized.

Pareto-ranking assigns the same cost to all non-dominated individuals, producing a long flat inverted ridge, as is shown in Figure 3. If achievable goals are specified, a discontinuity arises where solutions go from satisficing to unsatisfactory (Figure 4). A ridge, though shorter than in the previous case, is produced by those satisfactory solutions which are also non-dominated.

Giving one objective priority over the other considerably alters the landscape. In this case, the discontinuity corresponds to the transition from feasible to infeasible, and it happens to occur in the neighbourhood of the optimum (Figure 5). Finally, if both objectives are made into hard constraints, the feasible region becomes totally flat (Figure 6). This is because, in the absence of any other objectives, all solutions which satisfy both constraints must be considered equivalent.

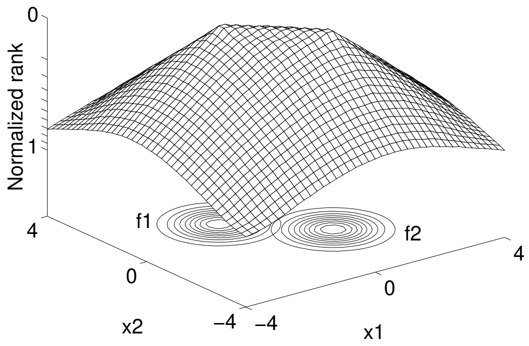


Figure 3: The cost landscape defined by Pareto-ranking (the contour plots are those of the individual objective functions f_1 and f_2).

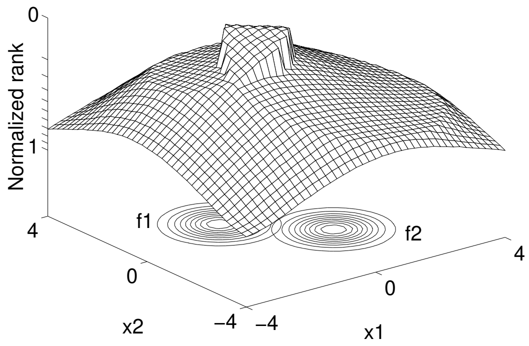


Figure 4: The effect of specifying two goals with the same priority.

Despite the underlying objectives being continuous, smooth and unimodal, the landscapes can be seen to exhibit features such as discontinuities, non-smoothness and flat regions. Optimizers capable of coping with such features are necessary for the decision making approach proposed to become useful, and EA-based optimizers are certainly eligible candidates.

6 Multiobjective Genetic Algorithms

The ranking of a population provides sufficient relative quality information to guide evolution. Given the current population ranking, different EAs will proceed with different selection and reproduction schemes, to produce a new set of individuals to be assessed.

This section will be concerned with the formulation of a Multiobjective Genetic Algorithm (MOGA), based on the ranking approach described earlier.

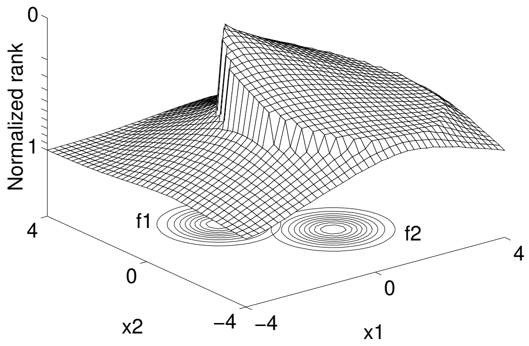


Figure 5: The effect of giving f_2 priority over f_1 (same goals).

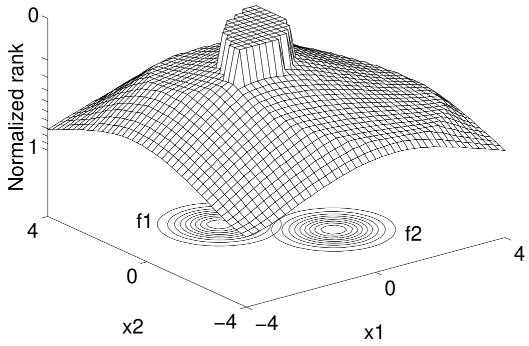


Figure 6: The effect of making both f_1 and f_2 into hard objectives (same goals).

6.1 Fitness assignment

Fitness is understood here as the number of offspring an individual is expected to produce through selection. It differs from individual utility, which reflects the result of the decision making process. The selection process determines which individuals actually influence the production of the next generation and is, therefore, a part of the search strategy.

The traditional rank-based fitness assignment is only slightly modified, as follows:

1. Sort population according to rank.

2. Assign fitness by interpolating from the best individual (rank $= 0$) to the worst (rank $= \max r^{(t)} < N$) according to some function, usually linear or exponential, but possibly of other type.

3. Average the fitness assigned to individuals with the same rank, so that all of them are sampled at the same rate while keeping the global population fitness constant.

Rank-based fitness assignment, as described, transforms the cost landscape defined by the ranks into a fitness landscape which is also independent from objective scaling.

6.2 Niche induction methods

In multimodal fitness landscapes, local optima offer the GA more than one opportunity for evolution. Although populations are potentially able to search many local optima, a finite population tends to settle on a single "good" optimum, even if other equivalent optima exist. This phenomenon is known as genetic drift, and has been well observed in natural, as well as artificial, evolution.

In the present case, where all non-dominated/preferred points are considered equally fit, the population of a GA can be expected to converge only to a small region of the trade-off surface, unless specific measures are taken against genetic drift [6, 21].

Niche induction methods [24] promote the simultaneous sampling of several different optima by favouring diversity in the population. Individuals tend to distribute themselves around the best optima, forming what is known as niches.

6.2.1 Fitness sharing

Fitness sharing [25] models individual competition for finite resources in a geographical environment. Individuals close to one another (according to some metric) mutually decrease each other's fitness. Even if initially considered less fit, isolated individuals are thus given a greater chance of reproducing, favouring diversification.

Finding a good trade-off description means achieving a diverse, if not uniform, sampling of the trade-off surface *in objective function space*. In the sharing scheme proposed here, share counts are computed based on individual distance in the objective domain, but only between individuals with the same rank. Sharing works by providing an additional selective pressure to that imposed by ranking, which counters the effects of genetic drift. Genetic drift becomes more important as more individuals in the population are assigned the same rank.

6.2.2 Setting the niche size

The sharing parameter σ_{share} establishes how far apart two individuals must be in order for them to decrease each other's fitness. The exact value which would allow a number of points to sample a trade-off surface whilst only tangentially interfering with one another depends on the area of such a surface. The following results assume that all objectives have the same, low priority, but can also be applied to a certain extent when there are multiple priority levels.

When expressed in the objective value domain, and due to the definition of non-inferiority, an upper limit for the size of the trade-off surface can be calculated from the minimum and maximum values each objective assumes within that surface. Let S be the trade-off set in the decision variable domain, $\mathbf{f}(S)$ the trade-off set in the objective domain and $\mathbf{y} = (y_1, \ldots, y_n)$ any objective vector in $\mathbf{f}(S)$. Also, let

$$\begin{aligned} \mathbf{m} &= (\min_{\mathbf{y}} y_1, \ldots, \min_{\mathbf{y}} y_n) = (m_1, \ldots, m_n) \\ \mathbf{M} &= (\max_{\mathbf{y}} y_1, \ldots, \max_{\mathbf{y}} y_n) = (M_1, \ldots, M_n) \end{aligned}$$

as illustrated in Figure 7.

The definition of non-dominance implies that any line parallel to any of the axes will have not more than one of its points in $\mathbf{f}(S)$, i.e., each objective is a single-valued function of the remaining objectives. Therefore, the true area of $\mathbf{f}(S)$ will be less than the sum of the areas of its projections according to each of the axes. Since the maximum area of each projection will be at most the area of the corresponding face of the hyperparallelogram defined by $\mathbf{m}$ and $\mathbf{M}$, the

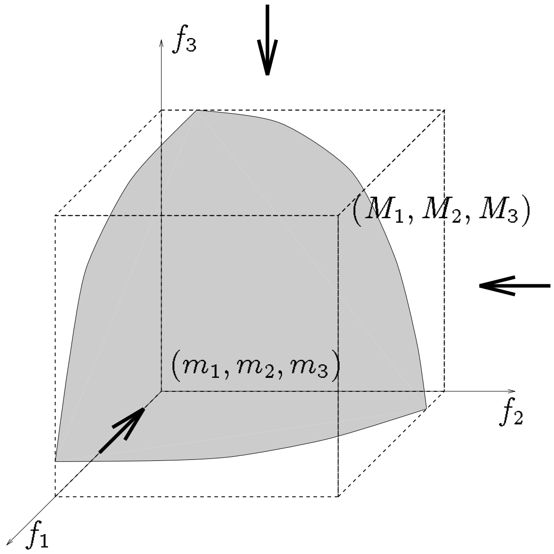


Figure 7: An example of a trade-off surface in 3-dimensional space

hyperarea of $\mathbf{f}(S)$ will be less than

$$A = \sum_{i=1}^{n} \prod_{\substack{j=1 \\ j \neq i}}^{n} \Delta_j$$

which is the sum of the areas of each different face of a hyperparallelogram of edges $\Delta_j = (M_j - m_j)$ (Figure 8).

The setting of σ_{share} also depends on how the distance between individuals is measured, and namely on how the objectives are scaled. In fact, the idea of sampling the trade-off surface *uniformly* implicitly refers to the scale in which objectives are expressed. The appropriate scaling of the objectives can often be determined as the aspect ratio which provides an *acceptable* visualization of the trade-off, or from the goal values. In particular, normalizing objectives by the best estimate of Δ_j available at each particular generation seems to yield good results (see the application examples in Part II [26]). This view is also expressed in a recent paper [27].

Assuming objectives are appropriately scaled, and using the ∞-norm as a measure of distance, the maximum number of points that can sample area A

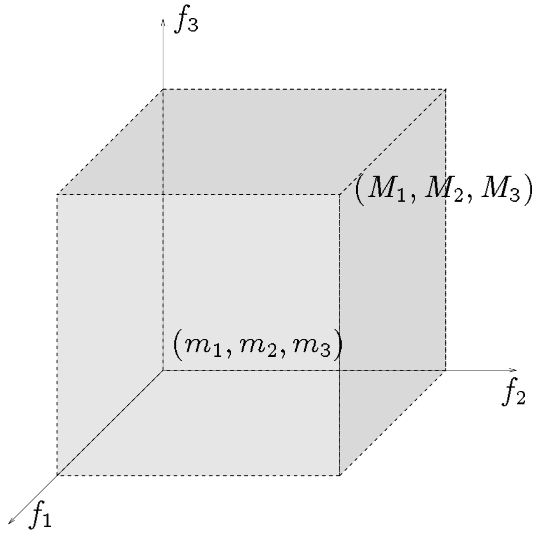


Figure 8: Upper bound for the area of a trade-off surface limited by the parallelogram defined by (m_1, m_2, m_3) and (M_1, M_2, M_3)

without interfering with each other can be computed as the number of hypercubes of volume σ_{share}^n that can be placed over the hyperparallelogram defined by A (Figure 9). This can be estimated from the difference in volume between two hyperparallelograms, one with edges $\Delta_i + \sigma_{\text{share}}$ and the other with edges Δ_i, by dividing it by the volume of a hypercube of edge σ_{share}, i.e.,

$$N = \frac{\prod_{i=1}^{n}(\Delta_i + \sigma_{\text{share}}) - \prod_{i=1}^{n}\Delta_i}{\sigma_{\text{share}}^n}$$

Conversely, given a number of individuals (points), N, it is possible to estimate σ_{share} by solving the $(n-1)$-order polynomial equation

$$N\sigma_{\text{share}}^{n-1} - \frac{\prod_{i=1}^{n}(\Delta_i + \sigma_{\text{share}}) - \prod_{i=1}^{n}\Delta_i}{\sigma_{\text{share}}} = 0$$

for $\sigma_{\text{share}} > 0$.

When there are objectives with different priorities and there are known solutions which meet all goals with priority higher than 1, trade-offs will involve

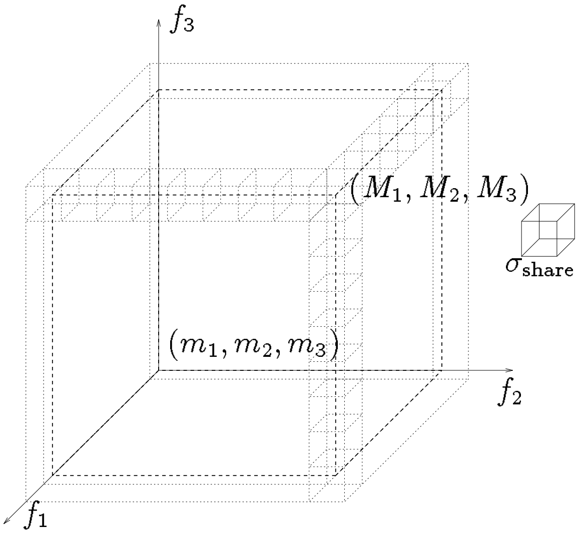


Figure 9: Sampling area A. Each point is σ_{share} apart from each of its neighbours (∞-norm)

only priority-1 objectives. The sharing parameter can, therefore, be computed for these only, using the expression above. This should be the case towards the end of the GA run in a problem where high-priority objectives can be satisfied.

Similarly, if the highest level of priority, i, which the preferred solutions, known at any given time, violate is greater than 1, the trade-offs explored by the preferability relation will not involve objectives with priority higher than i. Again, sharing may be performed while taking into account priority-i objectives only. It is a fact that objectives with priority lower than i *may* also become involved in the decision process, but this will only happen when comparing vectors with equal violating priority-i components. If this is the case, and the DM decides to move on to consider objectives with priority $i-1$, then the relevant priority-i objectives should either see their associated goals changed, or be associated priority $i-1$ by the DM for sharing to occur as desired.

6.2.3 Mating restriction

Mating restriction [24] tries to address the fact that individuals too different from each other are generally less likely than similar individuals to produce fit offspring through mating, by favouring the mating of similar individuals. In particular, the mating of distant members of the Pareto set can be expected to be inviable.

Mating restriction can be implemented much in the same way as sharing, by specifying how close individuals should be in order to mate. The corresponding parameter, σ_{mate}, can also be defined in the objective domain. After selection, one individual in the population is chosen, and the population searched for a mate within a distance σ_{mate}. If such an individual can be found, then mating is performed. Otherwise, a random individual is chosen [24].

Mating restriction assumes that neighbouring fit individuals are genotypically similar, so that they can form stable niches. Extra attention must therefore be paid to the coding of the chromosomes. Gray codes, as opposed to standard binary, are known to be useful for their property of adjacency. However, the coding of decision variables as the concatenation of independent binary strings cannot be expected to consistently express any relationship between them.

On the other hand, the Pareto set, when represented in the decision variable domain, will certainly exhibit such dependencies, as is the case in the example shown earlier in Figure 3. In that case, even relatively small regions of the Pareto-set may not be characterized by a single, high-order, schema and the ability of mating restriction to reduce the formation of lethals will be considerably diminished. As the size of the solution set increases, an increasing number of individuals is necessary in order to assure niche sizes small enough for the individuals within each niche to be sufficiently similar to each other.

Alternatively, the DM can reduce the size of the trade-off set by appropriately refining the current preferences. The GA must then be able to cope in some way with the corresponding change in the fitness landscape.

6.3 Progressive articulation of preferences

Setting aspiration levels in terms of goals and associated priorities is often difficult if done in the absence of any trade-off information. On the other hand, an accurate global description of the trade-off surface tends to be expensive, or even impossible to produce, since the Pareto set may not be bounded. Interactively refining preferences has the potential advantage of reducing computational effort by concentrating optimization effort on the region from which compromise solutions are more likely to emerge, while simultaneously providing the DM with trade-off information on which preference refinement can be based.

From the optimizer's point of view, the main difficulty associated with progressive articulation of preferences is the changing environment on which it must work. Consequently, the action of the DM may have to be restricted to the tightening of initially loose requirements, as with the moving-boundaries process [23]. In this case, although the overall optimization problem may change, the final solution must remain in the set of candidate solutions which satisfy the current preferences at any given time.

When EA-based optimizers are used, the DM may gain more freedom and actually decide to explore regions of the trade-off surface not considered in the initial set of preferences. The continuous introduction of a small number of random immigrants in the current population [28], for example, has been shown to improve the response of GAs to sudden changes in the objective function, while also potentially improving their performance as global optimizers.

Although there is no hard limit on how much the DM may wander away from preferences set originally, it must be noted that EAs will work on the utility function implicitly defined by the preference vectors the DM specifies. Any EA can only converge to a compromise solution if the DM comes to consistently prefer that solution to any others.

Giving the DM freedom to specify any preferences at any time also raises the question of what information should be stored during a run, so that no trade-off information acquired is lost. From Lemma 1, the non-dominated set of a particular problem contains at least one vector equivalent to any vector in the preferred set of the problem, defined by a given preference vector. Therefore, only the non-inferior individuals evaluated during a run of the algorithm need to be stored. A database of individuals currently non-dominated is also useful in setting the appropriate niche sizes for sharing and mating restriction, since it includes the relevant individuals from previous generations in the niche-size estimation process.

7 Concluding remarks

Soft objectives and constraints have been presented as individual aspects of a more general multi-function optimization problem. A decision making approach based on goal and priority information, which can be explored by evolutionary techniques such as genetic algorithms, has been formalized in terms of a transitive relation, here called preferability. The decision approach was then extended to the case where there are more than two alternatives to chose from, which also provided a means of visualizing the cost surfaces associated with the given decision approach over a search space.

Evolutionary algorithms, known to perform well on broad classes of ill-behaved problems, possess several properties desirable in a multiple objective optimizer. In particular, their simultaneous handling of multiple candidate solutions is well suited to the multiple solution character of most multiobjective problems. Mechanisms to promote diversity in the population were extended from the single-objective genetic algorithm with the generation of rich trade-off information in mind.

Trade-off information generated during a run of the algorithm can, in turn, be used to refine initial preferences until a suitable compromise solution is found.

Optimization effort may, in this way, be concentrated on the region of interest. The flexibility provided by EAs can also be explored at this level: on-line articulation of preferences implies non-stationary cost surfaces which the optimizer must handle satisfactorily.

Finally, the characterization of the multiobjective cost surfaces should prove useful in tailoring evolutionary algorithms to suit the needs of multiobjective optimization, such as the ability to handle ridges in the cost landscape in problems involving a large number of decision variables. However, standard GAs can already make good use of the preferability relation, as application examples presented in the second part of the paper [26] and elsewhere [29] demonstrate.

Acknowledgement

The first author gratefully acknowledges support by Programa CIENCIA, Junta Nacional de Investigação Científica e Tecnológica, Portugal. The authors also wish to acknowledge the support of the UK Engineering and Physical Sciences Research Council (Grant GR/J70857) in completion of this work.

References

[1] A. Ben-Tal, "Characterization of Pareto and lexicographic optimal solutions," in Fandel and Gal [30], pp. 1–11.

[2] C.-L. Hwang and A. S. M. Masud, *Multiple Objective Decision Making – Methods and Applications*, vol. 164 of *Lecture Notes in Economics and Mathematical Systems.* Berlin: Springer-Verlag, 1979.

[3] W. Dinkelbach, "Multicriteria decision models with specified goal levels," in Fandel and Gal [30], pp. 52–59.

[4] D. B. Fogel, *System Identification Through Simulated Evolution: A machine learning approach to modelling.* Needham, Massachusetts: Ginn Press, 1991.

[5] T. Bäck, F. Hoffmeister, and H.-P. Schwefel, "A survey of evolution strategies," in Belew and Booker [31], pp. 2–9.

[6] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, Massachusetts: Addison-Wesley, 1989.

[7] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, Massachusetts: MIT Press, 1992.

[8] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary Computation*, vol. 1, pp. 1–23, Spring 1993.

[9] Z. Michalewicz and C. Z. Janikow, "Handling constraints in genetic algorithms," in Belew and Booker [31], pp. 151–157.

[10] J. T. Richardson, M. R. Palmer, G. Liepins, and M. Hilliard, "Some guidelines for genetic algorithms with penalty functions," in Schaffer [32], pp. 191–197.

[11] D. Powell and M. M. Skolnick, "Using genetic algorithms in engineering design optimization with non-linear constraints," in Forrest [33], pp. 424–431.

[12] P. B. Wilson and M. D. Macleod, "Low implementation cost IIR digital filter design using genetic algorithms," in *IEE/IEEE Workshop on Natural Algorithms in Signal Processing*, vol. 1, (Chelmsford, U.K.), pp. 4/1–4/8, 1993.

[13] C. M. Fonseca, E. M. Mendes, P. J. Fleming, and S. A. Billings, "Non-linear model term selection with genetic algorithms," in *IEE/IEEE Workshop on*

Natural Algorithms in Signal Processing, vol. 2, (Essex, U.K.), pp. 27/1–27/8, 1993.

[14] W. Jakob, M. Gorges-Schleuter, and C. Blume, "Application of genetic algorithms to task planning and learning," in Männer and Manderick [34], pp. 291–300.

[15] D. Wienke, C. Lucasius, and G. Kateman, "Multicriteria target vector optimization of analytical procedures using a genetic algorithm. Part I. Theory, numerical simulations and application to atomic emission spectroscopy," *Analytica Chimica Acta*, vol. 265, no. 2, pp. 211–225, 1992.

[16] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in Grefenstette [35], pp. 93–100.

[17] M. P. Fourman, "Compaction of symbolic layout using genetic algorithms," in Grefenstette [35], pp. 141–153.

[18] C. M. Fonseca and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," Research report 527, Dept. Automatic Control and Systems Eng., University of Sheffield, Sheffield, U.K., July 1994.

[19] F. Kursawe, "A variant of evolution strategies for vector optimization," in *Parallel Problem Solving from Nature, 1st Workshop, Proceedings* (H.-P. Schwefel and R. Männer, eds.), vol. 496 of *Lecture Notes in Computer Science*, pp. 193–197, Berlin: Springer-Verlag, 1991.

[20] P. Hajela and C.-Y. Lin, "Genetic search strategies in multicriterion optimal design," *Structural Optimization*, vol. 4, pp. 99–107, 1992.

[21] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization," in Forrest [33], pp. 416–423.

[22] W. E. Hart and R. K. Belew, "Optimizing an arbitrary function is hard for the genetic algorithm," in Belew and Booker [31], pp. 190–195.

[23] V. Zakian and U. Al-Naib, "Design of dynamical and control systems by the method of inequalities," *Proceedings of the IEE*, vol. 120, no. 11, pp. 1421–1427, 1970.

[24] K. Deb and D. E. Goldberg, "An investigation of niche and species formation in genetic function optimization," in Schaffer [32], pp. 42–50.

[25] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms* (J. J. Grefenstette, ed.), pp. 41–49, Lawrence Erlbaum, 1987.

[26] C. M. Fonseca and P. J. Fleming, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms II: Application example," Research report 565, Dept. Automatic Control and Systems Eng., University of Sheffield, Sheffield, U.K., Jan. 1995.

[27] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," in *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, vol. 1, pp. 82–87, 1994.

[28] J. J. Grefenstette, "Genetic algorithms for changing environments," in Männer and Manderick [34], pp. 137–144.

[29] C. M. Fonseca and P. J. Fleming, "Multiobjective optimal controller design with genetic algorithms," in *Proc. IEE Control'94 International Conference*, vol. 1, (Warwick, U.K.), pp. 745–749, 1994.

[30] G. Fandel and T. Gal, eds., *Multiple Criteria Decision Making Theory and Application*, vol. 177 of *Lecture Notes in Economics and Mathematical Systems*. Berlin: Springer-Verlag, 1980.

[31] R. K. Belew and L. B. Booker, eds., *Genetic Algorithms: Proceedings of the Fourth International Conference*. San Mateo, CA: Morgan Kaufmann, 1991.

[32] J. D. Schaffer, ed., *Proceedings of the Third International Conference on Genetic Algorithms*. San Mateo, CA: Morgan Kaufmann, 1989.

[33] S. Forrest, ed., *Genetic Algorithms: Proceedings of the Fifth International Conference*. San Mateo, CA: Morgan Kaufmann, 1993.

[34] R. Männer and B. Manderick, eds., *Parallel Problem Solving from Nature, 2*. Amsterdam: North-Holland, 1992.

[35] J. J. Grefenstette, ed., *Genetic Algorithms and Their Applications: Proceedings of the First International Conference on Genetic Algorithms*, Lawrence Erlbaum, 1985.

A Proofs

A.1 Proof of Lemma 1

It suffices to show that

$$\mathbf{u} \;{}_{\mathrm{P}}\!\!<\; \mathbf{v} \underset{\mathbf{g}}{\Longrightarrow} (\mathbf{u}_{1,\ldots,i} \underset{\mathbf{g}_{1,\ldots,i}}{\prec} \mathbf{v}_{1,\ldots,i}) \vee (\mathbf{u}_{1,\ldots,i} \underset{\mathbf{g}_{1,\ldots,i}}{\equiv} \mathbf{v}_{1,\ldots,i})$$

for all $i = 1, \ldots, p$ and all $p \in I\!\!N$, which can be done by induction over i. The proof of the lemma is obtained by setting $i = p$.

Base clause $(i = 1)$

$$\mathbf{u} \;{}_{\mathrm{P}}\!< \mathbf{v} \;\underset{\mathbf{g}}{\Longrightarrow}\; (\mathbf{u}_1 \underset{\mathbf{g}_1}{\prec} \mathbf{v}_1) \;\vee\; (\mathbf{u}_1 \underset{\mathbf{g}_1}{\equiv} \mathbf{v}_1)$$

Proof

From Definition 1 (inferiority), if an n-dimensional vector $\mathbf{v}$ is inferior to another vector $\mathbf{u}$, then any component u_k of $\mathbf{u}$ will be less than or equal to the corresponding component v_k of $\mathbf{v}$, with $k = 1, \ldots, n$. This also implies that any subvector of $\mathbf{u}$ will either dominate or be equal to the corresponding subvector of $\mathbf{v}$. In particular, for $\mathbf{u}_1^{\overset{\mathbf{u}}{\frown}}$ and $\mathbf{v}_1^{\overset{\mathbf{u}}{\frown}}$,

$$\mathbf{u} \;{}_{\mathrm{P}}\!< \mathbf{v} \;\underset{\mathbf{g}}{\Longrightarrow}\; (\mathbf{u}_1^{\overset{\mathbf{u}}{\frown}} \;{}_{\mathrm{P}}\!< \mathbf{v}_1^{\overset{\mathbf{u}}{\frown}}) \vee (\mathbf{u}_1^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_1^{\overset{\mathbf{u}}{\frown}})$$

If $\mathbf{u}_1^{\overset{\mathbf{u}}{\frown}} \;{}_{\mathrm{P}}\!< \mathbf{v}_1^{\overset{\mathbf{u}}{\frown}}$, then, by Definition 3, $\mathbf{u}_1$ is preferable to $\mathbf{v}_1$. Otherwise, $\mathbf{u}_1^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_1^{\overset{\mathbf{u}}{\frown}}$, and, similarly, one can write:

$$\mathbf{u} \;{}_{\mathrm{P}}\!< \mathbf{v} \;\underset{\mathbf{g}}{\Longrightarrow}\; (\mathbf{u}_1^{\overset{\mathbf{u}}{\smile}} \;{}_{\mathrm{P}}\!< \mathbf{v}_1^{\overset{\mathbf{u}}{\smile}}) \vee (\mathbf{u}_1^{\overset{\mathbf{u}}{\smile}} = \mathbf{v}_1^{\overset{\mathbf{u}}{\smile}})$$

Again by Definition 3, if $\mathbf{u}_1^{\overset{\mathbf{u}}{\smile}} \;{}_{\mathrm{P}}\!< \mathbf{v}_1^{\overset{\mathbf{u}}{\smile}}$, then $\mathbf{u}_1 \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{v}_1$. Otherwise, $\mathbf{u}_1^{\overset{\mathbf{u}}{\smile}}$ is equal to $\mathbf{v}_1^{\overset{\mathbf{u}}{\smile}}$ and, by Definition 4, $\mathbf{u}_1$ is equivalent to $\mathbf{v}_1$.

Recursion clause $(1 < i \leq p)$

If

$$\mathbf{u} \;{}_{\mathrm{P}}\!< \mathbf{v} \;\underset{\mathbf{g}}{\Longrightarrow}\; (\mathbf{u}_{1,\ldots,i-1} \underset{\mathbf{g}_{1,\ldots,i-1}}{\prec} \mathbf{v}_{1,\ldots,i-1}) \;\vee\; (\mathbf{u}_{1,\ldots,i-1} \underset{\mathbf{g}_{1,\ldots,i-1}}{\equiv} \mathbf{v}_{1,\ldots,i-1})$$

then

$$\mathbf{u} \;{}_{\mathrm{P}}{<}\; \mathbf{v} \;\underset{\mathbf{g}}{\Longrightarrow}\; (\mathbf{u}_{1,\dots,i} \underset{\mathbf{g}_{1,\dots,i}}{\prec} \mathbf{v}_{1,\dots,i}) \;\vee\; (\mathbf{u}_{1,\dots,i} \underset{\mathbf{g}_{1,\dots,i}}{\equiv} \mathbf{v}_{1,\dots,i})$$

Proof

As before, one can write:

$$\mathbf{u} \;{}_{\mathrm{P}}{<}\; \mathbf{v} \;\underset{\mathbf{g}}{\Longrightarrow}\; (\mathbf{u}_i^{\overset{\mathbf{u}}{\frown}} \;{}_{\mathrm{P}}{<}\; \mathbf{v}_i^{\overset{\mathbf{u}}{\frown}}) \vee (\mathbf{u}_i^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_i^{\overset{\mathbf{u}}{\frown}})$$

If $\mathbf{u}_i^{\overset{\mathbf{u}}{\frown}} \;{}_{\mathrm{P}}{<}\; \mathbf{v}_i^{\overset{\mathbf{u}}{\frown}}$, then, by Definition 3, $\mathbf{u}_{1,\dots,i}$ is preferable to $\mathbf{v}_{1,\dots,i}$. If, on the other hand, $\mathbf{u}_i^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_i^{\overset{\mathbf{u}}{\frown}}$, then either $\mathbf{v}_i^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_i^{\overset{\mathbf{u}}{\smile}}$, in which case $\mathbf{u}_{1,\dots,i} \underset{\mathbf{g}_{1,\dots,i}}{\prec} \mathbf{v}_{1,\dots,i}$, or $\mathbf{v}_i^{\overset{\mathbf{u}}{\smile}} \leq \mathbf{g}_i^{\overset{\mathbf{u}}{\smile}}$.

In the latter case, and if the first alternative of the hypothesis is true, then $\mathbf{u}_{1,\dots,i}$ is preferable to $\mathbf{v}_{1,\dots,i}$. Otherwise, the second alternative of the hypothesis is that $\mathbf{u}_{1,\dots,i-1}$ is equivalent to $\mathbf{v}_{1,\dots,i-1}$. Since $\mathbf{u}_i^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_i^{\overset{\mathbf{u}}{\frown}}$ and $\mathbf{v}_i^{\overset{\mathbf{u}}{\smile}} \leq \mathbf{g}_i^{\overset{\mathbf{u}}{\smile}}$, the equivalence between $\mathbf{u}_{1,\dots,i}$ and $\mathbf{v}_{1,\dots,i}$ follows from Definition 4. □

A.2 Proof of Lemma 2

The transitivity of the preferability relation will be proved by induction over p. The proof will be divided into three parts, the first two of which apply to both the base clause ($p = 1$) and the recursion clause ($p > 1$). In the third part, the appropriate distinction between the two clauses is made.

Base clause ($p = 1$)

$$\mathbf{u}_{1,\dots,p} \underset{\mathbf{g}_{1,\dots,p}}{\prec} \mathbf{v}_{1,\dots,p} \underset{\mathbf{g}_{1,\dots,p}}{\prec} \mathbf{w}_{1,\dots,p} \underset{\mathbf{g}_{1,\dots,p}}{\Longrightarrow} \mathbf{u}_{1,\dots,p} \underset{\mathbf{g}_{1,\dots,p}}{\prec} \mathbf{w}_{1,\dots,p}$$

Recursion clause $(p > 1)$

If

$$\mathbf{u}_{1,\ldots,p-1} \underset{\mathbf{g}_{1,\ldots,p-1}}{\prec} \mathbf{v}_{1,\ldots,p-1} \underset{\mathbf{g}_{1,\ldots,p-1}}{\prec} \mathbf{w}_{1,\ldots,p-1} \underset{\mathbf{g}_{1,\ldots,p-1}}{\Longrightarrow} \mathbf{u}_{1,\ldots,p-1} \underset{\mathbf{g}_{1,\ldots,p-1}}{\prec} \mathbf{w}_{1,\ldots,p-1}$$

then

$$\mathbf{u}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{v}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{w}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\Longrightarrow} \mathbf{u}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{w}_{1,\ldots,p}$$

Proof

From Definition 3,

$$\begin{aligned}
\mathbf{u}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{v}_{1,\ldots,p} &\;\Rightarrow\; \mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \le \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}} \\
\mathbf{v}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{w}_{1,\ldots,p} &\;\Rightarrow\; \mathbf{v}_p^{\overset{\mathbf{v}}{\frown}} \le \mathbf{w}_p^{\overset{\mathbf{v}}{\frown}}
\end{aligned}$$

for all $p \ge 1$. On the other hand, since $\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} > \mathbf{g}_p^{\overset{\mathbf{u}}{\frown}}$,

$$\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \le \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}} \;\Rightarrow\; \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}} > \mathbf{g}_p^{\overset{\mathbf{u}}{\frown}}$$

which means that all components of $\mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}$ are also components of $\mathbf{v}_p^{\overset{\mathbf{v}}{\frown}}$ and, similarly for $\mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}$ and $\mathbf{w}_p^{\overset{\mathbf{v}}{\frown}}$. Therefore,

$$\mathbf{v}_p^{\overset{\mathbf{v}}{\frown}} \le \mathbf{w}_p^{\overset{\mathbf{v}}{\frown}} \;\Rightarrow\; \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}} \le \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}\,.$$

Case I: $\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \;{}_{\mathrm{P}}\!< \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}$

$$\left(\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \;{}_{\mathrm{P}}\!< \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}\right) \wedge \left(\mathbf{v}_p^{\overset{\mathbf{u}}{\frown}} \le \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}\right) \;\Rightarrow\; \mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \;{}_{\mathrm{P}}\!< \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}$$

which implies $\mathbf{u}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{w}_{1,\ldots,p}$, for all $p \ge 1$.

Case II: $(\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \wedge (\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}})$

$$(\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \wedge (\mathbf{v}_p^{\overset{\mathbf{u}}{\frown}} \leq \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}) \;\; \Rightarrow \;\; \mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \leq \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}$$

If $\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \;\mathrm{P}\!< \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}$, then $\mathbf{u} \underset{\mathbf{g}}{\prec} \mathbf{w}$.

If $\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}$, one must also note that $\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}}$ implies that there are at least some components of $\mathbf{v}_p^{\overset{\mathbf{v}}{\frown}}$ in $\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}}$, and similarly for $\mathbf{w}_p^{\overset{\mathbf{v}}{\frown}}$ and $\mathbf{w}_p^{\overset{\mathbf{u}}{\smile}}$. Consequently,

$$(\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}}) \wedge (\mathbf{v}_p^{\overset{\mathbf{v}}{\frown}} \leq \mathbf{w}_p^{\overset{\mathbf{v}}{\frown}}) \;\; \Rightarrow \;\; \mathbf{w}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}}$$

The preferability of $\mathbf{u}_{1,\ldots,p}$ over $\mathbf{w}_{1,\ldots,p}$ follows from

$$(\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}) \wedge (\mathbf{w}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}}) \tag{2}$$

for all $p \geq 1$.

Case III: $(\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \wedge (\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}} \leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}})$

In this case, $\mathbf{x}_p^{\overset{\mathbf{u}}{\frown}}$ and $\mathbf{x}_p^{\overset{\mathbf{u}}{\smile}}$ designate exactly the same vectors as $\mathbf{x}_p^{\overset{\mathbf{v}}{\frown}}$ and $\mathbf{x}_p^{\overset{\mathbf{v}}{\smile}}$, respectively, for $\mathbf{x} = \mathbf{u}, \mathbf{v}, \mathbf{w}, \mathbf{g}$. In the case where $\mathbf{v}_p^{\overset{\mathbf{v}}{\frown}} \;\mathrm{P}\!< \mathbf{w}_p^{\overset{\mathbf{v}}{\frown}}$, one can write:

$$(\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \wedge (\mathbf{v}_p^{\overset{\mathbf{u}}{\frown}} \;\mathrm{P}\!< \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}) \;\; \Rightarrow \;\; \mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} \;\mathrm{P}\!< \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}$$

which implies $\mathbf{u}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{w}_{1,\ldots,p}$, for all $p \geq 1$.

If $\mathbf{v}_p^{\overset{\mathbf{v}}{\frown}} = \mathbf{w}_p^{\overset{\mathbf{v}}{\frown}}$, then also $\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}$. If, in addition to that, $\mathbf{w}_p^{\overset{\mathbf{v}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{v}}{\smile}}$, one can write

$$(\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{w}_p^{\overset{\mathbf{u}}{\frown}}) \wedge (\mathbf{w}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}})$$

which implies that $\mathbf{u}_{1,\ldots,p}$ is preferable to $\mathbf{w}_{\mathbf{1},\ldots,\mathbf{p}}$ given $\mathbf{g}_{1,\ldots,p}$, for all $p \geq 1$.

If $\mathbf{w}_p^{\overset{\mathbf{v}}{\smile}} \leq \mathbf{g}_p^{\overset{\mathbf{v}}{\smile}}$, the base clause and the recursion clause must be considered separately.

Case III(a): $(p = 1)$

$$
\begin{aligned}
(\mathbf{u}_1 \underset{\mathbf{g}_1}{\prec} \mathbf{v}_1) \wedge (\mathbf{u}_1^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_1^{\overset{\mathbf{u}}{\frown}}) \wedge (\mathbf{v}_1^{\overset{\mathbf{u}}{\smile}} \leq \mathbf{g}_1^{\overset{\mathbf{u}}{\smile}}) \quad &\Rightarrow \quad (\mathbf{u}_1^{\overset{\mathbf{u}}{\smile}} \;{}_{\mathrm{P}}\!< \mathbf{v}_1^{\overset{\mathbf{u}}{\smile}}) \\
(\mathbf{v}_1 \underset{\mathbf{g}_1}{\prec} \mathbf{w}_1) \wedge (\mathbf{v}_1^{\overset{\mathbf{v}}{\frown}} = \mathbf{w}_1^{\overset{\mathbf{v}}{\frown}}) \wedge (\mathbf{v}_1^{\overset{\mathbf{v}}{\smile}} \leq \mathbf{g}_1^{\overset{\mathbf{v}}{\smile}}) \quad &\Rightarrow \quad (\mathbf{v}_1^{\overset{\mathbf{v}}{\smile}} \;{}_{\mathrm{P}}\!< \mathbf{w}_1^{\overset{\mathbf{v}}{\smile}}) \\
&\Rightarrow \quad (\mathbf{v}_1^{\overset{\mathbf{u}}{\smile}} \;{}_{\mathrm{P}}\!< \mathbf{w}_1^{\overset{\mathbf{u}}{\smile}})
\end{aligned}
$$

From the above, and given the transitivity of the inferiority relation, it follows that $\mathbf{u}_1^{\overset{\mathbf{u}}{\smile}} \;{}_{\mathrm{P}}\!< \mathbf{w}_1^{\overset{\mathbf{u}}{\smile}}$, which implies that $\mathbf{u}_1$ is preferable to $\mathbf{w}_1$ given $\mathbf{g}_1$, and proves the base clause.

Case III(b): $(p > 1)$

$$
\begin{aligned}
(\mathbf{u}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{v}_{1,\ldots,p}) \wedge (\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \wedge (\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}} \leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}}) \quad &\Rightarrow \quad (\mathbf{u}_{1,\ldots,p-1} \underset{\mathbf{g}_{1,\ldots,p-1}}{\prec} \mathbf{v}_{1,\ldots,p-1}) \\
(\mathbf{v}_{1,\ldots,p} \underset{\mathbf{g}_{1,\ldots,p}}{\prec} \mathbf{w}_{1,\ldots,p}) \wedge (\mathbf{v}_p^{\overset{\mathbf{v}}{\frown}} = \mathbf{w}_p^{\overset{\mathbf{v}}{\frown}}) \wedge (\mathbf{v}_p^{\overset{\mathbf{v}}{\smile}} \leq \mathbf{g}_p^{\overset{\mathbf{v}}{\smile}}) \quad &\Rightarrow \quad (\mathbf{v}_{1,\ldots,p-1} \underset{\mathbf{g}_{1,\ldots,p-1}}{\prec} \mathbf{w}_{1,\ldots,p-1})
\end{aligned}
$$

From the above, and if the hypothesis is true, then $\mathbf{u}_{1,\ldots,p-1} \underset{\mathbf{g}_{1,\ldots,p-1}}{\prec} \mathbf{w}_{1,\ldots,p-1}$, which implies that $\mathbf{u}_{1,\ldots,p}$ is preferable to $\mathbf{w}_{1,\ldots,p}$ given $\mathbf{g}_{1,\ldots,p}$, and proves the recursion clause. $\square$