Chapter 9
On Gradient-based Local Search to Hybridize Multi-objective Evolutionary Algorithms

Adriana Lara, Oliver Schütze and Carlos A. Coello Coello

Abstract Using evolutionary algorithms when solving multi-objective optimization problems (MOPs) has shown remarkable results during the last decade. As a consolidated research area it counts with a number of guidelines and processes; even though, their efficiency is still a big issue which lets room for improvements. In this chapter we explore the use of gradient-based information to increase efficiency on evolutionary methods, when dealing with smooth real-valued MOPs. We show the main aspects to be considered when building local search operators using the objective function gradients, and when coupling them with evolutionary algorithms. We present an overview of our current methods with discussion about their convenience for particular kinds of problems.

9.1 Introduction

Over the last few decades, a huge development on theoretical and practical approaches on solving multi-objective optimization problems (MOPs) has been done; either as multi-objective mathematical programming [14], or also as stochastic heuristics like evolutionary algorithms—named in this case as multi-objective evolutionary algorithms (MOEAs) [10, 9]. MOEAs are suitable to numerically approximate solutions of MOPs for several reasons; we can specially mention that, by nature, they spring an entire set of solutions on each run—instead of just one solution point as the traditional methods do. Having a set-oriented procedure is very

Adriana Lara
Mathematics Department ESFM-IPN, Edif. 9 UPALM, 07300 Mexico City, Mexico, e-mail: adriana@esfm.ipn.mx

Carlos A. Coello Coello, and Oliver Schütze
Computer Science Department, CINVESTAV-IPN, Av. IPN 2508, Col. San Pedro Zacatenco, 07360 Mexico City, Mexico e-mail: \{ccoello,schuetze\}@cs.cinvestav.mx

convenient when solving MOPs—we will clarify this idea later, using the next definitions.

Definition 9.1.1 *The multi-objective problem (MOP) is defined as minimizing*

$$F(x) := [f_1(x), f_2(x), \ldots, f_k(x)]^T \tag{9.1}$$

subject to:

$$g_i(x) \leq 0 \quad i = 1, 2, \ldots, m \tag{9.2}$$

where $x = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}^n$ is the vector of decision variables (or *decision parameters*), $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, ..., k$, are the objective functions and $g_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, ..., m$, are the constraint functions, which define the feasible region $\mathscr{X}$ of the problem. The problem above is also known as a *vector optimization* problem. If functions g_i are not present, we are dealing with an *unconstrained* MOP.

Solving a MOP is very different than solving a single-objective optimization problem (*i.e.* $k = 1$). Since some of the f_i are normally "in conflict" with each other,[1] the solution of a MOP is not given by a unique point; this is because normally no single solution exists that provides the best possible value for all the objectives. Consequently, solving a MOP implies finding a trade-off between all the objective functions; this requires the generation of a set of possible solutions instead of a single one—as in the single-objective optimization case. The notion of "optimality" that we just informally described, was originally proposed by Francis Ysidro Edgeworth in 1881 [13] and it was later generalized by Vilfredo Pareto, in 1896 [37]. This concept is known today as *Pareto optimality* and will be formally introduced next.

Definition 9.1.2 *Given two vectors $x, y \in \mathbb{R}^n$, we say that x* **dominates** *y (denoted by $x \prec y$) if $f_i(x) \leq f_i(y)$ for $i = 1, ..., k$, and $F(x) \neq F(y)$.*

Definition 9.1.3 *We say that a vector of decision variables $x \in \mathscr{X} \subseteq \mathbb{R}^n$ is* **non-dominated** *with respect to $\mathscr{X}$, if there does not exist another $x' \in \mathscr{X}$ such that $x' \prec x$.*

Definition 9.1.4 *We say that a vector of decision variables $x^* \in \mathscr{X} \subset \mathbf{R}^n$ is* **Pareto optimal** *if it is non-dominated with respect to $\mathscr{X}$.*

Definition 9.1.5 *a) The* **Pareto set** *$\mathscr{P}^*$ is defined by:*

$$\mathscr{P}^* = \{x \in \mathscr{X} \,|\, x \text{ is Pareto optimal}\}.$$

b) The **Pareto front** *$\mathscr{PF}^*$ is defined by:*

$$\mathscr{PF}^* = \{F(x) \in \mathbb{R}^k \mid x \in \mathscr{P}^*\}$$

[1] For example, one objective may refer to manufacture cost and another to quality of the product.

We thus wish to determine the Pareto optimal set, from the set $\mathscr{X}$ of all the decision variable vectors that satisfy the constraints of the problem. Note, however, that in practice just a finite representation of the Pareto optimal set is normally achievable.

Assuming x^* as a Pareto point of (9.1), there exist [27] a vector $\alpha \in \mathbb{R}^k$, with $0 \leq \alpha_i, i = 1, \ldots, k$ and $\sum_{i=1}^{k} = 1$ such that

$$\sum_{i}^{k} \alpha_i \nabla f_i(x^*) = 0. \tag{9.3}$$

A point x^* that satisfies (9.3) is called a **Karush-Kuhn-Tucker (KKT) point.**

Example 9.1. Consider the following unconstrained MOP:

$$\text{minimize } F(x,y) := \left[x^2 + y^2\,,\, (x-10)^2 + y^2\right]^T, \tag{9.4}$$

where $x, y \in \mathbb{R}$.

The Pareto set of this problem is the line segment $[(0,0),(10,0)] \subset \mathbb{R}^2$, and the Pareto front is shown, as a continuous line, in Figure 9.1.

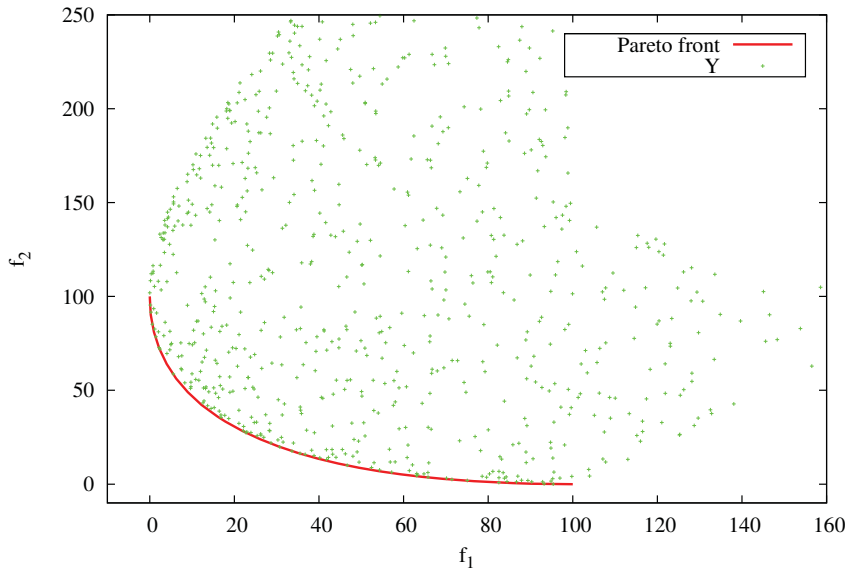

Fig. 9.1 This figure illustrates the Pareto front for the Example 9.1, the axes represent the value regarding each function—what we call the objective space. The points of Y are the images of randomly taken vectors inside the domain.

The set oriented nature of MOEAs had made them very popular among currently available methods for solving MOPs (see [8] for application examples). Even though, efficiency is the main drawback when they are used, since the evaluation of the objective functions is constantly made, each generation, for the whole population. Current research about improving MOEAs includes their hybridization with local searchers, in order to make a guided search at a particular moment of the procedure. The coupling of evolutionary algorithms with any local search procedure is also known as a *memetic algorithm* [35].

Local search operators depend on the domain, since the goal is precisely to take advantage of previous knowledge of the problem. A lot of research has been done on local searchers, and memetic MOEAs, for discrete and combinatorial optimization problems (see for example [25, 24, 26]). For multi-objective problems with continuous domains, the local search has not been deeply studied as itself; this is probably due to the fact that neighborhoods on continuous spaces have an infinite cardinality. The vicinity notion in continuous domain problems is related with open balls or manifolds, so the natural choice is to study them by the differential properties of the functions; this leads to the use of gradient information to compute better solutions.

In this chapter we focus on local searchers, for continuous MOPs, built to use (implicitly and/or explicitly) the gradient information of the objective functions. We focus mainly on unconstrained cases, but give some insights for the extension to constrained search spaces.

In the rest of this chapter we develop the main ideas from particular work [31, 28, 32, 29, 30] on local searchers for continuous memetic algorithms. We introduce in the next section the basic concepts to study the gradient geometry in the case of MOPs. In Section 9.3 we present algorithms, based on multi-objective line search, emphasizing their particular features. The applicability of these methods, when hybridizing MOEAs, is presented in Section 9.4 as well as some discussion over the main hybridization aspects. Finally some conclusions and possible extensions are included in Section 9.5.

9.2 Descent Cones and Directions

When solving an optimization problem with several objective functions, we have to deal with several gradients—one for each function. Some gradient-based MOEAs have been built to perform descent movements alternating the single gradient of each objective function [16, 21]; this could have certain applications on preferences management. However, when designing a special local searcher for continuous multi-objective problems, our focus lies on finding a suitable mechanism to improve solutions using simultaneously every function gradient. The reason is that such a good local searcher must have the feature that after its application, over a single solution x_i, it throws a new solution x_{i+1} which is better in the Pareto sense (*i.e.* x_{i+1} *dominates* x_i). This feature is very important when analyzing the global convergence of the methods.

Using gradient information to guide the simultaneous descent of several functions can not be done in a straightforward way. The main reason is that, since the objective functions are each other in conflict, their gradients typically point toward different directions. This turns the task of finding a common improving direction into another multi-objective problem. To understand the local behavior of gradient-based methods in multi-objective optimization, descent cones constitute the main tool since they were first used in the multi-objective evolutionary context by Brown and Smith [6, 7].

We state next the main concepts and introduce the notation required for further discussion.

Let $f_1,\ldots,f_k : \mathbb{R}^n \to \mathbb{R}$ be continuous and differentiable, and $\langle\cdot,\cdot\rangle$ denote the standard inner product in $\mathbb{R}^n$.

Let

$$\nabla f_i(x) = \left(\frac{\partial f_i(x)}{\partial x_1},\ldots,\frac{\partial f_i(x)}{\partial x_n}\right)$$

be the gradient of the function f_i at x. Then, for each $x \in \mathbb{R}^n$ and each $i \in \{1,\ldots,k\}$, with $\nabla f_i(x) \neq 0$, we define:

$$H_{x,i} = \left\{ v \in \mathbb{R}^n : \left\langle \nabla f_i(x), v \right\rangle = 0 \right\},$$

$$H^+_{x,i} = \left\{ v \in \mathbb{R}^n : \left\langle \nabla f_i(x), v \right\rangle \geq 0 \right\},$$

and

$$H^-_{x,i} = \left\{ v \in \mathbb{R}^n : \left\langle \nabla f_i(x), v \right\rangle \leq 0 \right\}.$$

Since the set $H_{x,i}$ is the orthogonal complement of the vector $\nabla f_i(x)$, it is (if $\nabla f_i(x) \neq 0$) in general a hyperplane[2] of $\mathbb{R}^n$; also, it divides the space in two n–dimensional sets: $H^+_{x,i}$ and $H^-_{x,i}$ (see Figure 9.2).

Definition 9.2.1 *We denote*

$$C_x(-,-,\ldots,-) = \bigcap_{i=1}^{k} H^-_{x,i.} \setminus \{0\}.$$

as the descent cone *pointed at x (see Figure 9.3). Similarly, the* ascent cone *is defined by* $C_x(+,+,\ldots,+) = \bigcap_{i=1}^{k} H^+_{x,i.} \setminus \{0\}$, *and the* diversity cones *are the intersection of hyperplanes when they are not all of the form* $H^+_{x,i}$ *and neither all of them of the form* $H^-_{x,i}$.

When having k functions in a MOP, each function f_i determines a gradient ∇f_i and a hyperplane $H_{x,i}$ for a certain solution x. Summarizing, for each point x in the search space, the k functions determine a division of this space into one descent

[2] A hyperplane of an n-dimensional space is a subspace with dimension $n1$.

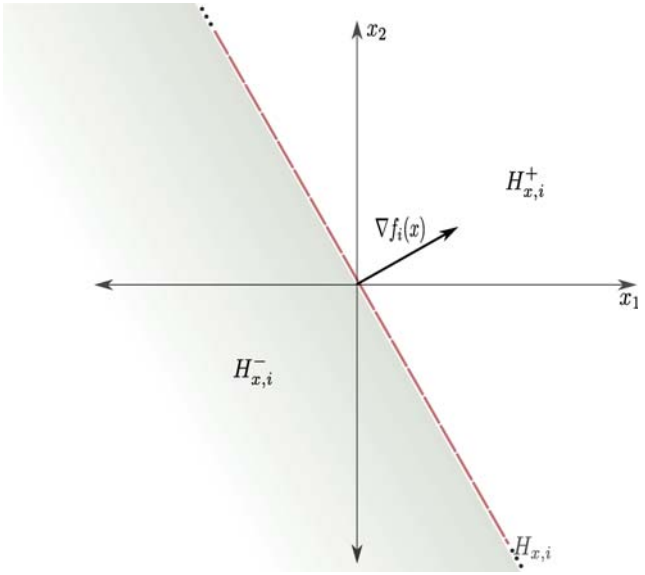

Fig. 9.2 This figure shows the division of $\mathbb{R}^2$ into the two half-spaces, $H^+_{x,i}$ and $H^-_{x,i}$, induced by the gradient of the function f_i at the point x.

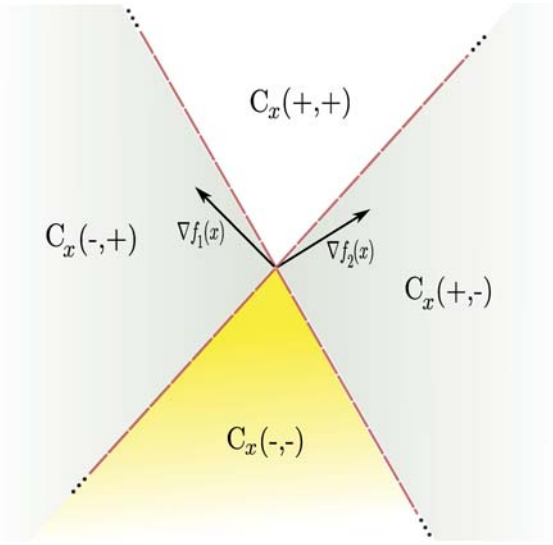

Fig. 9.3 This figure shows the ascent cone $C_x(+,+)$, the descent cone $C_x(-,-)$, and the diversity cones $C_x(+,-)$ and $C_x(-,+)$, for a certain point x, for a bi-objective problem.

cone, one ascent cone and $2^k - 2$ diversity cones—as they were previously named in [6][3].

Descent cones describe the dynamics of the search from a gradient-based geometry point of view, and this has several important implications (see for example the results presented in [42]) that are useful when building local search operators for algorithms over real numbers. One primary observation [6] is that when the point x is far away from any Pareto optimal point the descent cone is big. This happens because the gradients are almost aligned; then, the chance of randomly generating a direction/solution which simultaneously improves all the functions is high (nearly 50%). On the other hand, when the point x is near a Pareto optimal point, the gradients are almost linearly dependent, which means that the descent cone shrinks, and the possibility of randomly generating a better point is low (see Figure 9.4).

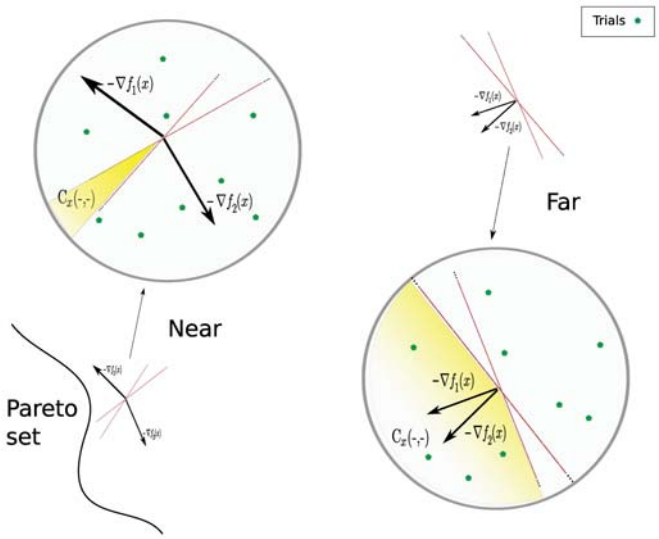

Fig. 9.4 This figure shows two cases, when the point is near and when it is far from a Pareto set; the descent cone $C_x(-,-)$ shrinks down when the Pareto set is reached.

The above analysis could explain why MOEAs have good performance at the beginning of the search, and a slow convergence rate at latter stages—when points are near to the Pareto front and the chance of generating randomly better points is reduced. This observation inspires guidelines for a suitable hybridization; if it is possible to identify when the evolutionary search is no longer producing good results, this is then the time when the gradient-based local search can take part of the process—in order to certainly improve solutions in a deterministic way.

[3] In [6] the descent cones are defined, for illustrative purposes, by pictures of the corresponding affine hyperplanes described here. We are stating the formal definitions using no affine hyperplanes just to be consistent with our approach.

Improving solutions implies performing movements in specific search directions. In multi-objective optimization, as we have noticed, these directions should be able to throw (at least locally) better solutions regarding all the functions simultaneously; this is formally said by the next definition.

Definition 9.2.2 *A vector $v \in \mathbb{R}^n$ is called a multi-objective descent direction of the point $x \in \mathbb{R}^n$ if*

$$v \in C_x(-,-,\ldots,-).$$

In other words, a multi-objective descent direction is such that the directional derivatives with respect to v in x are non-positive, *i.e.* $\langle \nabla f_i(x), v \rangle \leq 0$ for all $i \in 1,\ldots,m$ without allowing them to be all equal to zero. This means that if we perform a small movement over v, we obtain a local improvement (decrease) simultaneously for all the objective functions. In the next section, we present different ways to calculate descent directions and to perform movements toward (and along) the Pareto set. In particular, we present a result for the construction of a descent direction in the simplest multi-objective case—two objectives.

9.3 Practical Approaches

9.3.1 Movements Toward the Optimum

When running a MOEA, new points are generated by variation operators in order to move (evolve) the population of solutions toward an approximation of the Pareto set. Due to the stochastic nature of the process, a probability to generate non-improving points does exist. On the other hand, when using gradient information within a MOEA, the resulting hybrid algorithm is able to perform directed accurate movements toward an improved solution. This newly computed point dominates the original one when multi-objective descent search directions—together with a suitable step size control—are used. As we mentioned before, computing such directions is also a multi-objective problem [2]; since each objective provides its own (gradient-based) range of movements for descent, all of these possible directions need to be properly combined into a single one in order to efficiently guide a MOEA.

9.3.1.1 A Simple Bi-objective Descent Direction

The simplest way to combine two gradients, in order to get a common descent direction, is by adding them. This fact has been already observed (e.g. [11]) but it has not been exploited in memetic algorithms yet. The next result shows that this operation throws, in fact, a bi-objective descend direction. Unfortunately, Proposition 9.1 cannot be generalized for more than two objective functions. For that case, applying other approaches—mentioned next—is necessary to obtain the descent direction; even when this represents a higher computational cost.

Proposition 9.1. *Let $x \in \mathbb{R}^n$, and let $f_1, f_2 : \mathbb{R}^n \to \mathbb{R}$ define a bi-objective MOP. Then, the direction*

$$\overline{\nabla}_x = -\left(\frac{\nabla f_1(x)}{||\nabla f_1(x)||} + \frac{\nabla f_2(x)}{||\nabla f_2(x)||}\right), \tag{9.1}$$

where $||\cdot|| = ||\cdot||_2$, is a descent direction at x for the MOP.

Proof: Let us denote $\nabla_i := \frac{\nabla f_i(x)}{||\nabla f_i(x)||}$ for $i = \{1,2\}$, and θ be the angle between ∇_1 and ∇_2. Then

Similarly, $\langle \overline{\nabla}_x, \nabla_2 \rangle \leq 0$; then, $\overline{\nabla}_x$ is a descent direction of the point x for the defined MOP. Note that if $\nabla f_i(x) = 0$, for $i = 1$ or $i = 2$, then x is a KKT point. □

One of the main issues when using gradient-based tools is how to validate the increase of the computational cost, after using the method, with the achieved improvements. The currently available MOEAs that use descent directions as local search engines have two sources of computational cost. The first one is associated to the fitness function evaluations required to estimate the gradients and to perform the line search. The second source is related to the computation of the descent direction itself. In this sense, and unlike previous approaches [15, 2, 20], this way (Equation 9.1) of calculating a direction has the advantage of having a zero cost for the computation of the descent direction. We claim that this procedure is the simplest way to combine the gradients of two functions, but it can not be generalized to more than two functions, since a similar arithmetic combination of them does not produce a descent direction in general, see the following example:

Example 9.2. Assuming a three-objective problem such that, for a certain x,

$$\begin{aligned}
\nabla f_1(x) &= (1.000, 1.000, 1.000) \\
\nabla f_2(x) &= (-0.944, 0.970, 0.374) \\
\nabla f_3(x) &= (0.836, -0.177, -0.334).
\end{aligned}$$

Then, computing

$$\begin{aligned}
\overline{\nabla}_x &= -\left(\frac{\nabla f_1(x)}{||\nabla f_1(x)||} + \frac{\nabla f_2(x)}{||\nabla f_2(x)||} + \frac{\nabla f_3(x)}{||\nabla f_3(x)||}\right) \\
&= (-0.3826, -0.5262, -0.3730)
\end{aligned}$$

leads to

$$\begin{aligned}
\langle \overline{\nabla}_x, \nabla_1 \rangle &= -1.2818 < 0. \\
\langle \overline{\nabla}_x, \nabla_2 \rangle &= 0.4423 > 0. \\
\langle \overline{\nabla}_x, \nabla_3 \rangle &= -0.2889 < 0.
\end{aligned}$$

with $\nabla_i := \frac{\nabla f_i(x)}{||\nabla f_i(x)||}$ for $i = \{1,2,3\}$. Then $\overline{\nabla}_x$ is not a common descent direction.

9.3.1.2 General Approaches

The most commonly used proposals for computing multi-objective descent directions are the one from Fliege and Svaiter [15] and the one from Schäffler *et al.* [38]. Theoretical results about their efficacy and convergence are available. These methods return a common descent direction for all the objectives after solving a quadratic optimization problem—which could be derived into a linear one. These approaches have been already incorporated into multi-objective memetic strategies [44], [28], [31]. Their main drawback is that, being greedy methods used to perform descents, they can present a bias in case of unbalanced magnitudes of the gradient vectors. However, these methods present some advantages, like having an intrinsic stopping criterion and being quite effective in practice.

As an unbiased alternative, it is possible to use normalized gradients and to work with the proposals developed by Bosman and DeJong [2], and, to manage constraints, by Harada *et al.* [20]. These methods look into the entire Pareto set of descent directions and choose (randomly, at each step) one direction within it. The cost of this approach is again related to solving a system of linear equations.

It is worth noting that other approaches [7, 31], which do not explicitly compute the gradients, have also an acceptable performance using less resources. They are completely based on the information extracted from the descent cones, at a particular moment during the search. One of these methods will be presented in the next section.

9.3.2 Movements Along the Pareto Set

Descent directions are not the only interesting directions to move along during the multi-objective optimization search. Sometimes it is also beneficial to perform movements along the Pareto set; or also, specifically directed movements toward a particular region. Moving in a direction along the Pareto set is also possible using gradient-based continuation methods such as those described in [22, 41, 19], or those with no estimation of the gradients required, as in [7] and [31]. We describe next an operator which is able to perform movements either toward or along the Pareto set, making an automatic switch between these two types of movements when a KKT point is almost[4] reached.

9.3.2.1 The Hill Climber with Side-step

In [31] a novel point-wise iterative search procedure, called the Hill Climber with Side-step (HCS) has been proposed. This procedure is designed to perform local search over continuous domain MOPs. Based on the descent cone size and the

[4] We set a tolerance parameter to consider if an approximation is ‘good enough’.

Karush-Kuhn-Tucker conditions [27], the HCS is capable of moving both toward or along the set of (local) Pareto points, depending on the location of the current iterate. Since proximity and a good distribution of solutions are features for the approximations of Pareto sets, this local search operator has shown potential when combined with MOEAs.

Two variants of the HCS have been proposed, a gradient-free version (denoted as HCS1) and a version that exploits explicit gradient information (denoted as HCS2). Both of them can be used as standalone algorithms to explore parts of the Pareto set, starting with one single solution, and are able to handle constraints of the model to some extent. In the following we will explain the two approaches as standalone algorithms, complemented with the recommendations to be used within a memetic algorithm.

HCS1 is based on the observation that the objective gradients are practically aligned when the initial point is far away from the optima—and the descent cone is almost equal to the half-spaces associated with each objective. So, the procedure starts with an initial point x_0, and the next iterate x_1 is randomly chosen from a vicinity $B(x_0,r)$ with radius r. If $x_1 \prec x_0$ the movement direction for improvement is set as $v = x_1 - x_0$; if $x_0 \prec x_1$, then the direction is flipped.

When a solution is near to a Pareto point, the gradients point nearly toward opposite directions, and the probability of generating a dominated or a dominating point—like in the case above—is low (see Figure 9.4). So, when $\tilde{x}_1$ is not comparable against x_0, the point is stored, and labeled, as a point which corresponds to a specific diversity cone; then, a new trial point is generated. After N_{nd} trials obtaining mutually non-dominated solutions, the proximity with the optima is assumed and this triggers a 'sidestep' movement over the Pareto front.

To perform this sidestep movement, the stored points $\tilde{x}_1, \ldots \tilde{x}_{N_{nd}}$ are used in the following way. If $\tilde{x}_1 - x_0$ is, for example, in the cone $C(+,-)$, then $x_0 - \tilde{x}_1$ is in the opposite cone $C(-,+)$ (for the bi-objective MOPs, the general k-objective case is analogue). When the limit for unsuccessful trials is reached, a search along $C(-,+)$ is performed; Taking advantage of the accumulated information, the following direction is used:

$$v_{acc} = \frac{1}{N_{nd}} \sum_{i=1}^{N_{nd}} s_i \frac{\tilde{x}_i - x_0}{\|\tilde{x}_i - x_0\|}, \tag{9.2}$$

where

$$s_i = \begin{cases} 1 & \text{if } f_1(\tilde{x}_i) < f_1(x_0). \\ \text{-1} & \text{else.} \end{cases} \tag{9.3}$$

By construction, v_{acc} is in $C(-,+)$, and by averaging the trial search directions we aim to obtain a direction[5] which is ideally 'perpendicular' to the (small) descent cone. Note that in this case v_{acc} is indeed a 'sidestep' to the upward movement of the hill climbing process as desired, but this search direction does not necessarily have to point along the Pareto set (see next subsection for an alternative method with better guidance properties); also, there is no guarantee that v_{acc} indeed points

[5] This direction has previously been proposed as a local guide for a multi-objective particle swarm algorithm in [5].

to a diversity cone, but in other case, there will be an improvement on the solution anyway. This means that, even with these two considerations, this sidestep is still a good option in practice, when working with few objectives or when coupling the operator with evolutionary methods.

Algorithm 9.1: HCS1 (without Using Gradient Information)

Require: starting point $x_0 \in Q$, radius $r \in \mathbb{R}^n_+$, number N_{nd} of trials, MOP with $k = 2$
Ensure: sequence $\{x_l\}_{l \in \mathbb{N}}$ of candidate solutions
1: $a := (0,\ldots,0) \in \mathbb{R}^n$
2: $nondom := 0$
3: **for** $l = 1,2,\ldots$ **do**
4: set $x_l^1 := x_{l-1}^b$ and choose $x_l^2 \in B(x_l^1, r)$ at random
5: choose $i_0 \in \{1,2\}$ at random
6: **if** $x_l^1 \prec x_l^2$ **then**
7: $v_l := x_l^2 - x_l^1$
8: compute $t_l \in \mathbb{R}_+$ and set $\tilde{x}_l^n := x_l^2 + t_l v_l$.
9: choose $x_l^b \in \{\tilde{x}_l^b, x_l^1\}$ such that $f(x_l^b) = \min(f(\tilde{x}_l^n), f(x_l^1))$
10: $nondom := 0, \quad a := (0,\ldots,0)$
11: **else if** $x_l^2 \prec x_l^1$ **then**
12: proceed analogous to case "$x_l^1 \prec x_l^2$" with
13: $v_l := x_l^1 - x_l^2$ and $\tilde{x}_l^n := x_l^1 + t_l v_l$.
14: **else**
15: **if** $f_{i_0}(x_l^2) < f_{i_0}(x_l^1)$ **then**
16: $s_l := 1$
17: **else**
18: $s_l := -1$
19: **end if**
20: $a := a + \frac{s_l}{N_{nd}} \frac{x_l^2 - x_l^1}{\|x_l^2 - x_l^1\|}$
21: $nondom := nondom + 1$
22: **if** $nondom = N_{nd}$ **then**
23: compute $\tilde{t}_l \in \mathbb{R}_+$ and set $\tilde{x}_l^n := x_l^1 + \tilde{t}_l a$.
24: $nondom := 0, \quad a := (0,\ldots,0)$
25: **end if**
26: **end if**
27: **end for**

Algorithm 9.1 shows the pseudocode of the HCS1 operator as a standalone process. The sidestep direction is determined by the value of i_0 (see line 5 and lines 15-20). For simplicity, the value of i_0 is chosen at random. In order to introduce an orientation to the search, the following modifications can be done in the bi-objective case: in the beginning, i_0 is fixed to 1 for the following iteration steps. When the sidestep (line 23) has been performed N_s times during the run of an algorithm, this indicates that the current iteration is already near to the (local) Pareto set, and this vector is stored in x_p. If in the following no improvements can be achieved according to f_1 within a given number N_i of sidesteps, the HCS 'jumps' back to x_p, and a similar process is started but aiming for improvements according to f_2. That is, i_0 is set to -1 for the following steps (see Figure 9.5). A possible stopping criterion,

hence, could be to stop the process when no improvements can be achieved according to f_2 within another N_i sidesteps along $C(+,-)$. This is in fact used as stopping criterion. Finally, for the computation of t_l more attention has to be paid (see Section 9.3.4). In the original proposal, a strategy analog to [12] was used for the presented experiments.

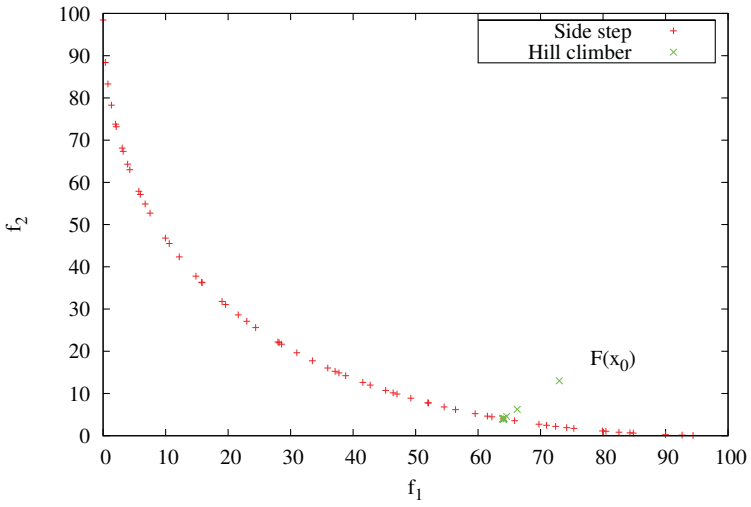

Fig. 9.5 This figure shows the performance of the HCS1 as standalone algorithm for Example 9.1; the 'anchor' picture shows the entire Pareto front, built using the steering mechanism previously described.

The second version of the HCS consists of a movement toward the Pareto front, using either a gradient-based descent direction, or a continuation-based movement in case this descent direction does not exist. A possible realization of the HCS2 is by using the descent direction presented in [38] (or the one in [15] as an alternative), which is described next.

Let a MOP be given and $q : \mathbb{R}^n \to \mathbb{R}^n$ be defined by

$$q(x) = \sum_{i=1}^{k} \hat{\alpha}_i \nabla f_i(x), \tag{9.4}$$

where $\hat{\alpha}$ is a solution of

$$\min_{\alpha \in \mathbb{R}^k} \left\{ \| \sum_{i=1}^{k} \alpha_i \nabla f_i(x) \|_2^2 ; \alpha_i \geq 0, i = 1, \ldots, k, \sum_{i=1}^{k} \alpha_i = 1 \right\}. \tag{9.5}$$

Then either $q(x) = 0$ or $-q(x)$ is a descent direction for all objective functions $f_1, \ldots, f_k$ in x. This states that for every point $x \in Q$ which is not a KKT–point a descent direction (i.e., a direction where all objectives' values can be improved) can be found by solving the quadratic optimization problem (9.5). In case $q(x) = 0$ the point x is a KKT–point. Thus, a test for optimality is automatically performed when computing the descent direction for a given point $x \in Q$. Given such a point x, the quadratic optimization problem (9.5) can be solved leading to the vector $\hat{\alpha}$. In case

$$\| \sum_{i=1}^{k} \hat{\alpha}_i \nabla f_i(x) \|_2^2 \geq \varepsilon_{\mathscr{P}}, \tag{9.6}$$

i.e., if the square of the norm of the weighted gradients is larger than a given threshold $\varepsilon_{\mathscr{P}} \in \mathbb{R}_+$, the candidate solution x can be considered to be 'away' from a local Pareto point, and thus, it makes sense to seek for a dominating solution. For this, the descent direction (9.4) can be taken together with a suitable step size control. If the value of the term in (9.6) is less than $\varepsilon_{\mathscr{P}}$, this indicates that x is already in the vicinity of a local Pareto point. In that case one can integrate elements from (multi-objective) continuation [22, 1] to perform a search along the Pareto set. For simplicity we assume that we are given a KKT–point $\hat{x}$ and the respective weight $\hat{\alpha}$ obtained by (9.5). Then the point $(\hat{x}, \hat{\alpha}) \in \mathbb{R}^{n+k}$ is contained in the zero set of the auxiliary function $\tilde{F} : \mathbb{R}^{n+k} \to \mathbb{R}^{n+1}$ of the given MOP which is defined as follows:

$$\tilde{F}(x, \alpha) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i \nabla f_i(x) \\ \sum_{i=1}^{k} \alpha_i - 1 \end{pmatrix}. \tag{9.7}$$

In [22] it has been shown that the zero set $\tilde{F}^{-1}(0)$ can be linearized around $\hat{x}$ by using a QU-factorization of $\tilde{F}'(\hat{x}, \hat{\alpha})^T$, i.e., the transposed of the Jacobian matrix of $\tilde{F}$ at $(\hat{x}, \hat{\alpha})$. To be more precise, given a factorization

$$\tilde{F}'(\hat{x}, \hat{\alpha})^T = QU \in \mathbb{R}^{(n+k)\times(n+k)}, \tag{9.8}$$

where $Q = (Q_N, Q_K) \in \mathbb{R}^{(n+k)\times(n+k)}$ is orthogonal with $Q_N \in \mathbb{R}^{(n+k)\times(n+1)}$ and $Q_K \in \mathbb{R}^{(n+k)\times(k-1)}$, the column vectors of Q_K form—under some mild regularity assumptions on $\tilde{F}^{-1}(0)$ at $(\hat{x}, \hat{\alpha})$, see [22]—an orthonormal basis of the tangent space of $\tilde{F}^{-1}(0)$. Hence, it can be expected that each column vector $q_i \in Q_K$, $i = 1, \ldots, k-1$, points (locally) along the Pareto set and is thus well suited for a sidestep direction. The step size control is explained in detail in [31].

Based on the above discussion, the HCS2 is presented in Algorithm 9.2. It is worth to remark that this is one possible realization and that there exist certainly other ways leading, however, to similar results. For instance, alternatively to the descent direction used in Algorithm 9.2, the ones proposed in [15] and [4] can be taken as well. The threshold $\varepsilon_{\mathscr{P}}$ is used for the vicinity test of a given local Pareto

point. This value is certainly problem dependent, but can be made 'small' due to convergence properties of the hill climber (e.g., [15]).

Further, the movement along the Pareto set can be realized by predictor-corrector methods [22, 1] which consist, roughly speaking, of a repeated application of a predictor step obtained by a linearization of $\tilde{F}^{-1}(0)$ and a corrector step which can be done, e.g. via a Gauss-Newton method.
It is worth noting that although the HCS2 is proposed for the unconstrained case, an extension to the constrained case for the hill climber is possible (see, e.g., [15] for possible modifications); but, this is not straightforward for the movement along the Pareto set (i.e., the sidestep). Though it is possible to extend system (9.7) using equality constraints (e.g., by introducing slack variables to transform the inequality constraints into equality constraints); but according to [22], this could lead to efficiency problems in the numerical treatment.

Algorithm 9.2: HCS2 (Using Gradient Information)

Require: starting point $x_0 \in Q$
Ensure: sequence $\{x_l\}_{l\in\mathbb{N}}$ of candidate solutions
for $l = 0,1,2,\ldots$ **do**
  compute the solution $\hat{\alpha}$ of (9.5) for x_l.
  if $\|\sum_{i=1}^{k} \hat{\alpha}_i \nabla f_i(x_l)\|_2^2 \geq \varepsilon_{\mathscr{P}}$ **then**
    $v_l := -q(x_l)$
    compute $t_l \in \mathbb{R}_+$ and set $x_{l+1} := x_l + t_l v_l$
  else
    compute $\tilde{F}'(\hat{x},\hat{\alpha})^T = (Q_N, Q_K)U$ as in (9.8)
    choose a column vector $\tilde{q} \in Q_K$ at random
    compute $\tilde{t}_l \in \mathbb{R}_+$ and set $x_{l+1} := x_l + \tilde{t}_l \tilde{q}$.
  end if
end for

The HCS shows large potential when used within multi-objective memetic algorithms; mainly because it performs an efficient local search which starts with one point and ends not only with an improvement of this point, but also, with two candidates for spread. Figure 9.6 ilustrates the application of the HCS as a local searcher over a population of three individuals, when the points are far, a hill climber movement (HC) is perfomed; and the hill climber sith side step (HCS) is applied when the optima is 'almost' reached. The operator can repeat the descent step—hill climber—several times until a sidestep is triggered. An analysis of the performance of these two versions, and comparisons in terms of efficiency and cost, can be found in [31].

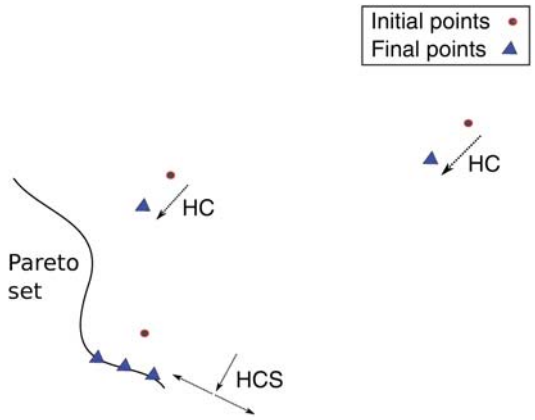

Fig. 9.6 This figure shows the way to use the HCS as local search operator, in order to be coupled with a set oriented heuristic. The local search starts with a point and ends with an improved point, or with three points, the one obtained by the gradient-based descent and the other two obtained by sidesteps.

9.3.3 Directed Movements

It is possible, furthermore, to perform directed movements not only toward and along the Pareto front, but also to any desired direction in the objective space; for this, the Directed Search Method (DS) was recently proposed [40, 39]. This method has the advantage of performing movements along determined paths in the objective function space.

9.3.3.1 The Directed Search Method

When working with MOPs, performing local search movements toward a particular region is sometimes desired. In this scope, a proposal [40] for the computation of directed search movements was recently introduced. The complexity of this operator is again linear and only first order gradient information is necessary in order to use it. This approach calculates a gradient-based descent direction using a controlled bias toward regions of interest determined in objective space; because of that, this proposal has many potential applications in the context of designing hybrid MOEAs.

Under the assumption that $x_0 \in \mathbb{R}^n$ is not a KKT point—in particular is also not a Pareto optimal point—a descent direction must exist such that all the directional derivatives must be non positive. In other words, once a vector $-\alpha \in \mathbb{R}^k$ is chosen,

with $\alpha \in \mathbb{R}^k, 0 \leq \alpha, \sum_{i=1}^{k} \alpha_i = 1$, representing a desired search direction in image space. Then, a search direction $v \in \mathbb{R}^n$ in parameter space is sought such that for $y_0 = x_0 + tv$, where $t \in \mathbb{R}_+$ is the step size, it holds:

$$\lim_{t \searrow 0} \frac{f_i(y_0) - f_i(x_0)}{t} = \langle \nabla f_i(x_0), v \rangle = -\alpha_i, \quad i = 1, \ldots, k. \tag{9.9}$$

Using the Jacobian J of F,

$$J(x_0) = \begin{pmatrix} \nabla f_1(x_0)^T \\ \vdots \\ \nabla f_k(x_0)^T \end{pmatrix} \in \mathbb{R}^{k \times n}, \tag{9.10}$$

Equation (9.9) can be stated in matrix vector notation as

$$J(x_0)v = -\alpha. \tag{9.11}$$

Then, this search direction v can be computed by solving a system of linear equations. It is important to remark that system (9.11) is typically highly underdetermined—since in most cases the number of parameters is (much) higher than the number of objectives in a given MOP—which implies that its solution is not unique.

To find a solution of this system, one option is to take the greedy choice, i.e., the solution with the smallest norm, which is given by

$$v = J(x_0)^+(-\alpha), \tag{9.12}$$

where $J(x_0)^+ \in \mathbb{R}^{n \times k}$ denotes the pseudo inverse of $J(x_0)$. In case the rank of $J := J(x_0)$ is maximal (which we will assume in the following), the pseudo inverse is given by $J^+ = J^T(JJ^T)^{-1}$. Given a 'descent direction' $-\alpha \in \mathbb{R}^k$, a sequence of dominating points in 'direction $-\alpha$' can thus be found by numerically solving the following initial value problem:

$$\begin{aligned} x(0) &= x_0 \in \mathbb{R}^n \\ \dot{x}(t) &= J(x(t))^+(-\alpha), \quad t > 0. \end{aligned} \tag{IVP$_{(-\alpha)}$}$$

One observation worth noting is that even if $-\alpha$ is a descent direction, there is no guarantee that the solution curve c of (IVP$_{(-\alpha)}$) always leads to a Pareto optimal solution. When the image $F(\mathbb{R}^n)$ is bounded below, however, c leads to a boundary point x^* of the image. Since for x^* it holds $rank(J(x^*)) < k$, this can be used to trace numerically the end point of (IVP$_{(-\alpha)}$) in a certain way—the numerical integration can be stopped if the condition number $\kappa_2(J(x_i))$ exceeds a given (large) threshold. In [34], more insights about efficient computation of such end points, e.g. specialized predictor-corrector (PC) methods are presented. For the case of m active inequality constraints, the details can be found in [40, 33].

Even when setting α looks like imposing a weights vector, this method is able to also reach non-convex regions on the front; this is illustrated in Figure 9.7. For

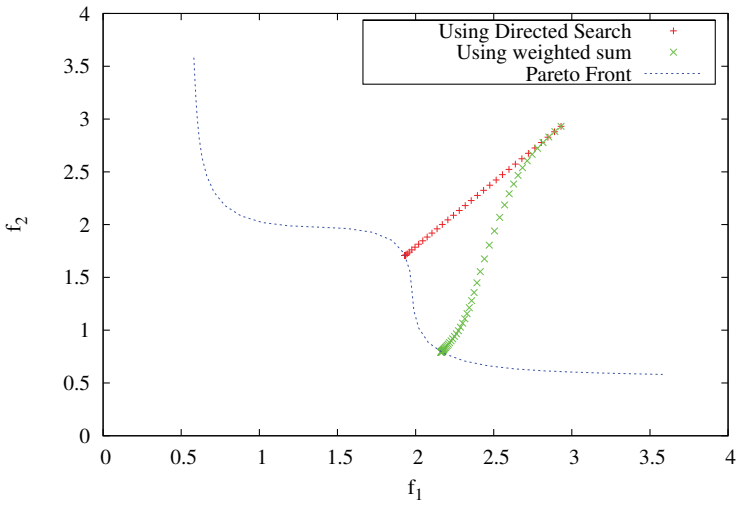

Fig. 9.7 Numerical result and comparison for the Directed Search method and the weighted sum method, starting from the same point and performing the movement with direction $\alpha = [0.55, 0.45]^T$, toward a non-convex region.

several reasons, this method would be a good choice to be used as a local engine inside a gradient-based memetic algorithm. For example, it is a good alternative to greedy approaches like [15] and [38]; it can also perform a movement similar to the one proposed in [4], but in a user-preference controlled way. This means that, using a reference point $Z \in \mathbb{R}^k$ we can compute α as follows

$$\alpha(x_0, Z) := \frac{F(x_0) - Z}{||F(x_0) - Z||_1},$$

having in this way a guided route for the descent (see Figure 9.8). When applying greedy strategies, for a multi-objective gradient-based descent, for functions with unbalanced magnitudes, it is possible to get an undesired bias. This method avoids this potential problem. This method also gives us the best direction (nearest point) according to the reference point Z, by applying the following continuation strategy to perform a search along the Pareto set as follows:

Assume we are given a (local) Pareto point x and the related KKT weight α, i.e., such that

$$\sum_{i=1}^{k} \alpha_i \nabla f_i(x) = 0 \tag{9.13}$$

and further we assume that

$$rank(J(x)) = k - 1 \tag{9.14}$$

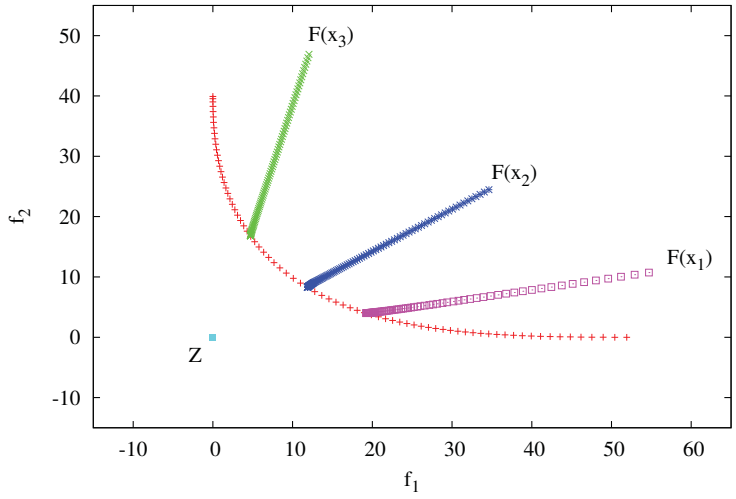

Fig. 9.8 Solution paths thrown by the Directed Search method for three different starting points, using the same target point Z.

It is known (e.g., [22]) that in this case α is orthogonal to the Pareto front at $y = F(x)$, i.e.,

$$\alpha \perp T_y \partial F(\mathbb{R}^n), \tag{9.15}$$

where $\partial F(\mathbb{R}^n)$ denotes the border of the image $F(\mathbb{R}^n)$. Thus, a search orthogonal to α (again in image space) could be promising to predict a new solution near x along the Pareto set. To use (9.11), for instance a QR-factorization of α can be computed, i.e.,

$$\alpha = QR, \tag{9.16}$$

where $Q = (q_1, \ldots, q_k) \in \mathbb{R}^{k \times k}$ is an orthogonal matrix and q_i, $i = 1, \ldots, k$, its column vectors, and $R = (r_{11}, 0 \ldots, 0)^T \in \mathbb{R}^{k \times 1}$ with $r_{11} \in \mathbb{R} \setminus \{0\}$ (for the computation of such a factorization we refer e.g. to [36]). Since by Equation (9.16) $\alpha = r_{11} q_1$, i.e., $\alpha \in span\{q_1\}$, and Q orthogonal it follows that the column vectors $q_2, \ldots, q_k$ build an orthonormal basis of the hyperplane which is orthogonal to α. Thus, a promising well-spread set of search directions v_i may be the ones which satisfy

$$J(x) v_i = q_i, \quad i = 2, \ldots, k. \tag{9.17}$$

In a next step, the predicted points $p_i = x + t_i v_i$, where $t_i \in \mathbb{R}$ are step sizes, can be corrected back to the Pareto set. For this, one can e.g. solve numerically (IVP$_{(-\alpha)}$) using p_i as initial value and $-\alpha$ as direction in (IVP$_{(-\alpha)}$). Continuing this procedure iteratively leads to a particular PC variant for MOPs. Note that here no Hessians of the objectives have to be computed which is indeed the case for other existing multi-

objective PC methods.

To conclude, we mention that when mixing local searchers with evolutionary algorithms, having a method to steer the search—like the Directed Search method—has a lot of potential. We will show this in the next section.

9.3.4 Step-length Computation

As a final remark for the section, we note that once the movement direction is set, choosing a suitable step size is not an easy task in multi-objective optimization. In practice, different proposals have been tested with good results [12, 43, 31]. One possibility [15] is also to adapt the well known Armijo-Goldstein rule to the multi-objective case, and accept any step length t that holds

$$F(x+tv) \leq F(x) \; - \; ct\,J(x)\,v,$$

where $F : \mathbb{R}^n \to \mathbb{R}^k$ is the multi-objective function and $J(x) : \mathbb{R}^n \to \mathbb{R}^n$ is the Jacobian matrix of F at x. The value $c \in (0,1)$ is a control parameter to decide how fine grained, numerically speaking, the descent will be. A bad choice of c can highly increase the cost related to function evaluations. With a suitable initial step length, this method is easily applicable; however, finding an efficient approach in the general multi-objective case is still an open problem.

9.4 Toward the Hybridization

9.4.1 Main Issues

We have shown, in the previous sections, that there are some options available to compute directions—based on gradient information—in the context of multi-objective optimization; but, the question of how to efficiently integrate them into a population-based context—as in the set oriented algorithms, such as MOEAs are—remains wide open. In this sense, it is also worth noting that the suitable choice of the movement direction relies also on the location of the point, and on the location of all the other population individuals (see Figure 9.9).

Once the descent direction v, for a specific point $x \in \mathbb{R}^n$, is obtained, the new line search function is defined as

$$\begin{aligned} f_v^i : \mathbb{R} &\longrightarrow \mathbb{R} \\ t &\longmapsto f_i(x+tv). \end{aligned}$$

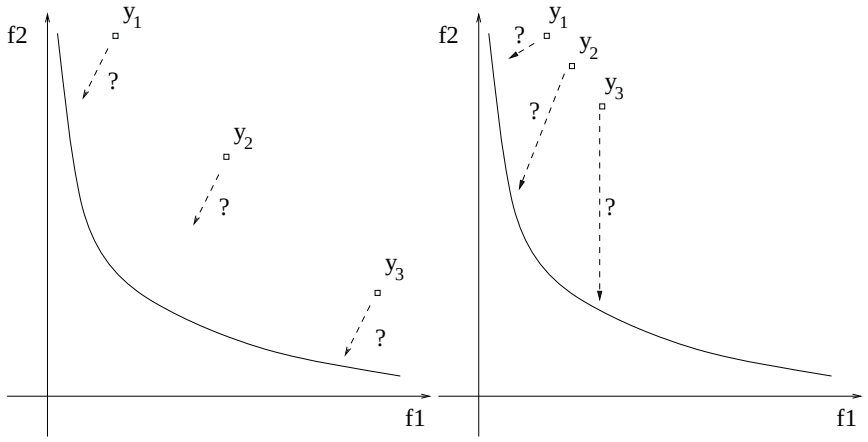

Fig. 9.9 This figure shows that, in a population context, an efficient search direction for each individual depends on several factors.

Now, the difficulty turns into the computation of the step size for f_v^i at x, because it is again a multi-objective problem (see Figure 9.10). Even when approximations of the optimal step size for each function are easy to estimate, the question is how to combine this information to find a common step to maximize the improvement. In this case an exact step size calculation is not possible; but, the use of inexact methods, like those described at Section 9.3.4, is a good option in practice. Step size is an important issue since it compromises the efficiency of the local search and the memetic algorithm as well. Even when the computation of the search direction and the step length are apparently independent of each other, a bad choice of the second can raise the cost of the procedure by several times.

Talking about hybridization with gradient-based local searchers, another important issue is that it is not possible to *a priori* determine a specific amount of resources, to be specifically devoted, for the local search procedure and the global one as well. In this sense, in order to produce efficient algorithms, an adaptive mechanism to control the use of local search is advised; but, this is itself a non-trivial problem. Gradient-based local search is typically a high-cost procedure; then, such a balance mechanism must be capable of determining when the gradient method outperforms the pure evolutionary search, during the solution of a specific problem—which means that the procedure is cost-effective. One possible option is to incorporate local search, as a method to refine solutions, only at the end of the search (as suggested in [18]); but this leads to a two-stage algorithm, and the precise time to start the local search is critical. In this sense, a proposal about doing the switch during running-time is presented in [29]. The main idea is to start the second stage when the evolutionary procedure is not improving the solutions anymore; for example, when all the individuals are mutually non-dominated, and the selection mechanism of the MOEA faces troubles because of that. In this case a

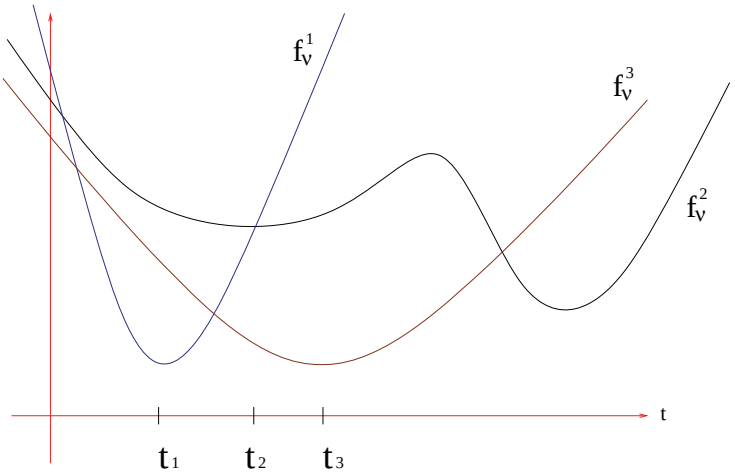

Fig. 9.10 Simultaneous line search for three functions along the direction *v*. Even when solving separately the *m* line searches getting t_1, t_2 and t_3, it is not possible to say which is the suitable step length for all of them together.

refinement with a certain direction for improvements is desirable—precisely what is done with gradient-based methods. On the other hand, using deterministic search directions—over the stochastic technique—may accelerate the 'convergence' of the search when dealing with problems with very smooth fitness landscapes; then, in this cases the use of gradient-based procedures from the beginning of the search can be advantageous [28]. In conclusion, an adaptive switch mechanism between the two search engines (the local and the global one) during running-time is the desirable choice; but this is still an open research problem.

A final consideration comes with the fact that when dealing with population-based algorithms, it is important to keep a bounded archive of improved solutions. When using evolutionary algorithms, there are several available mechanisms to limit the size of this archive. This turns into an issue when resources have been spent using local search, since this bounding mechanisms typically delete solutions without notice if it has been previously improved by an expensive mechanism or not. When using gradient information at the end of the search, solutions are accurate in proportion to the amount of resources we want to spend in the line search. Finally, when building hybrid algorithms, saving the previously refined solutions from the truncation mechanism is mandatory; then, it is necessary to set special mechanisms to archive solutions in these cases.

9.4.2 Early Hybrids

Early attempts to combine MOEAs with gradient-based information use well-known MOEAs as their baseline algorithms, and simply replace the mutation operator by a line search procedure [44][45]. Other proposals have used gradient-based local search as an additional operator to be applied under certain rules during the MOEA run [2][3]. Within this same line of thought, other mathematical programming techniques such as SQP [17] or the reference point method [49] have also been coupled with MOEAs [23][46]. It is worth noting for completeness, that in hybrid MOEAs using separately the gradients of single objectives is also possible, only when dealing with few-objective problems [21].

In [28] it is presented a two-stage algorithm (GBMES) which uses gradient-based line search in a first stage, in order to quickly reach points that are close to the Pareto set. During this stage, a MOEA with a reduced population size was combined with the gradient-based local search. The balance of resources to apply the local search was naturally given by the selection of the best individuals, from the small population. After spending a certain amount of resources, the second stage attempted to reconstruct the front (see [28] for details). This proposal has the natural drawbacks of being a two-stage procedure and has some limitations to be applied in general. Nevertheless, this work showed the potential advantages of using a descent direction of movement when dealing with problems with a high number of parameters; mostly because when the space is highly dimensional, the evolutionary techniques—avid to keep a uniform distribution of the population—can easily get lost, making very profitable to count with certain search directions.

In [31] the HCS operator is coupled with two state-of-the-art MOEAs. The effectiveness of this hybrid algorithm was assessed in conventional test problems for MOEAs. In most cases, the advantages of the hybrid method over the original MOEA were very clear. The balance of the resources for this local search operator was made through an *a priori* set probability function. We confirmed (previously stated in [24]) with this work that the part of the balance (local vs. global search) is, in general, the most important issue in terms of efficiency, for this type of algorithms.

Also, an attempt to perform a dynamic balance control (between the two operators) is presented in [29]. Here, an indicator over the improvements made by the evolutionary search is combined with a probability function, which controls the number of individuals to be modified by the local search. This approach shows promising results on traditional benchmarks.

9.5 Conclusions and New Trends

We presented, in this chapter, different gradient-based local search operators with diverse features. Apart from the cost of computing the gradients of the objective functions, we described a computational zero-cost descent direction, suitable for

bi-objective problems. We also presented operators to perform movements both toward and along the Pareto set. The HCS has been presented in two versions, with and without explicit use of gradient information, and its main feature lies on the automatic switch between the two movements (hill climbing and sidestep) which makes it a powerful local searcher.

The terms *convergence* and *spread* are commonly used when talking about approximations of sets in the multi-objective context (mostly the Pareto set or its image, the Pareto front). Convergence is about the proximity of the solutions toward the set of interest, while spread relates to the minimal distance of these solutions to each other—which should be maximized in order to 'capture' as much as possible from the set of interest. The feature of our methods, to operate between a range of movements—toward, along and directed—is very promising when solving MOPs, because of the importance of the balance between convergence and spread. Even when these method are compromised, like other gradient based local searchers, when used in problems with a high number of local Pareto points, they have been found to be efficient in combination with MOEAs.

Although particular descent directions, to improve the convergence of approximation sets, have been suggested for MOPs (e.g., [15, 38]), their use within memetic strategies is not widely accepted. This is maybe due to an undesired bias of the chosen descent directions. The presented DS method goes beyond and allows the search to be steered in a particular controlled direction, which has so far not been considered. Here, the greedy direction from a given solution can be redefined according to preferences—in order to steer the search along all the regions of the front, or those that are difficult to explore by conventional MOEA mechanisms.

Regarding open research problems, we can state the adaptation of inexact methods for step-size control (such as Wolfe conditions, Armijo conditions, etc.) to the multi-objective search. Ensuring convergence, and the study of speed of convergence, for these methods are important issues to address when building an efficient interleaving between MOEAs and local search. We also mention that adaptive control of the resources allowed for the local search during runtime is one of the main issues of this hybridization. This control should automatically determine when the evolutionary operators are not producing improvements and when the introduction of gradient-based local search is cost effective. Another promissing possibility is to combine several local search heuristics in a same hybrid algorithm by an adaptive control mechanism, like in [47, 48].

Finally, a very important aspect of hybrid MOEAs is the archive management. It is not desirable that our archiving strategy (like crowding or truncation) destroys the refinement previously done to certain solutions. Hence, every gradient-based algorithm should be coupled with a suitable archiving strategy. Interleaving the selection process and the gradient-based improvements with the archiving strategy of a MOEA is also a promising path for future research.

Acknowledgements

The first author acknowledges scholarship support from CONACyT as a Ph.D. student, and from IPN project no. 20121478. The second author acknowledges support from CONACyT project no. 128554. The third author acknowledges support from CONACyT project no. 103570.

References

1. Eugene L. Allgower and Kurt Georg. *Numerical Continuation Methods*. Springer Verlag, 1990.
2. Peter A.N. Bosman and Edwin D. de Jong. Exploiting Gradient Information in Numerical Multi-Objective Evolutionary Optimization. In Hans-Georg Beyer et al., editor, *2005 Genetic and Evolutionary Computation Conference (GECCO'2005)*, volume 1, pages 755–762, New York, USA, June 2005. ACM Press.
3. Peter A.N. Bosman and Edwin D. de Jong. Combining Gradient Techniques for Numerical Multi-Objective Evolutionary Optimization. In Maarten Keijzer et al., editor, *2006 Genetic and Evolutionary Computation Conference (GECCO'2006)*, volume 1, pages 627–634, Seattle, Washington, USA, July 2006. ACM Press. ISBN 1-59593-186-4.
4. Peter A.N. Bosman and Dirk Thierens. The Naive MIDEA: A Baseline Multi-objective EA. In Carlos A. Coello Coello, Arturo Hernández Aguirre, and Eckart Zitzler, editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, pages 428–442, Guanajuato, México, March 2005. Springer. Lecture Notes in Computer Science Vol. 3410.
5. Jürgen Branke and Sanaz Mostaghim. About selecting the personal best in multi-objective particle swarm optimization. In *Parallel Problem Solving from Nature - PPSN IX*, pages 523–532, Springer Berlin / Heidelberg, 2006.
6. Martin Brown and R.E. Smith. Effective use of directional information in multi-objective evolutionary computation. In *Genetic and Evolutionary Computation GECCO 2003*, volume Volume 2723/2003 of *Lecture Notes in Computer Science*, pages 778–789. Springer Berlin / Heidelberg, 2003.
7. Martin Brown and R.E. Smith. Directed Multi-objective Optimization. *International Journal of Computers, Systems and Signals*, 6(1):3–17, 2005.
8. Carlos A. Coello Coello and Gary B. Lamont, editors. *Applications of Multi-Objective Evolutionary Algorithms*. World Scientific, Singapore, 2004. ISBN 981-256-106-4.
9. Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York, second edition, September 2007. ISBN 978-0-387-33254-3.
10. Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.
11. Michael Dellnitz, Oliver Schütze, and T. Hestermeyer. Covering Pareto Sets by Multilevel Subdivision Techniques. *Journal of optimization theory and applications*, 124(1):113–136, 2005.
12. J.E. Dennis and R.B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, 1983.
13. F. Y. Edgeworth. *Mathematical Physics*. P. Keagan, London, England, 1881.
14. Mattias Ehrgott and M.M. Wiecek. *Multiple criteria decision analysis: state of the art surveys*, volume 78, chapter Multiobjective Programming, pages 667–722. Springer Verlag, 2005.
15. Jörg Fliege and Benar Fux Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, 51(3):479–494, February 2000.

16. P. García, M.L. Fátima, C.F. Julián, C.C. Rafael, A. Carlos, and A.G. Hernández-Díaz. Hibridación de métodos exactos y heurísticos para el problema multiobjetivo. *Rect@*, Actas15(1), 2007.
17. P.E. Gill, W. Murray, and M.A. Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002.
18. Ken Harada, Kokolo Ikeda, and Shigenobu Kobayashi. Hybridizing of Genetic Algorithm and Local Search in Multiobjective Function Optimization: Recommendation of GA then LS. In Maarten Keijzer et al., editor, *2006 Genetic and Evolutionary Computation Conference (GECCO2006)*, volume 1, pages 667–674, Seattle, Washington, USA, July 2006. ACM Press. ISBN 1-59593-186-4.
19. Ken Harada, J. Sakuma, S. Kobayashi, and I. Ono. Uniform sampling of local pareto-optimal solution curves by pareto path following and its applications in multi-objective GA. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, page 820. ACM, 2007.
20. Ken Harada, Jun Sakuma, and Shigenobu Kobayashi. Local Search for Multiobjective Function Optimization: Pareto Descent Method. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 659–666, New York, NY, USA, 2006. ACM Press.
21. Alfredo G. Hernández-Díaz, Carlos A. Coello Coello, Fátima Pérez, Rafael Caballero, Julián Molina, and Luis V. Santana-Quintero. Seeding the Initial Population of a Multi-Objective Evolutionary Algorithm using Gradient-Based Information. In *2008 Congress on Evolutionary Computation (CEC'2008)*, pages 1617–1624, Hong Kong, June 2008. IEEE Service Center.
22. Claus Hillermeier. *Nonlinear Multiobjective Optimization: A Generalized Homotopy Approach*, volume 135 of *International Series of Numerical Mathematics*. Birkhäuser, 2001.
23. Xiaolin Hu, Zhangcan Huang, and Zhongfan Wang. Hybridization of the Multi-Objective Evolutionary Algorithms and the Gradient-based Algorithms. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, volume 2, pages 870–877, Canberra, Australia, December 2003. IEEE Press.
24. 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.
25. Andrzej Jaszkiewicz. Genetic local search for multiple objective combinatorial optimization. *European Journal of Operational Research*, 137(1):50–71, 2002.
26. Joshua D. Knowles. *Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization*. PhD thesis, The University of Reading, Department of Computer Science, Reading, UK, January 2002.
27. H. W. Kuhn and A. W. Tucker. Nonlinear Programming. In *Proceedings of the Second Berkeley Symposium on Mathematics Statistics and Probability*. University of California Press, 1951.
28. Adriana Lara, Carlos A. Coello Coello, and Oliver Schütze. Using Gradient-Based Information to Deal with Scalability in Multi-objective Evolutionary Algorithms. In *2009 IEEE Congress on Evolutionary Computation (CEC'2009)*, pages 16–23, Trondheim, Norway, May 2009. IEEE Press.
29. Adriana Lara, Carlos A. Coello Coello, and Oliver Schütze. A painless gradient-assisted multiobjective memetic mechanism for solving continuous bi-objective optimization problems. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, IEEE Press, 2010.
30. Adriana Lara, Carlos A. Coello Coello, and Oliver Schütze. Using gradient information for multi-objective problems in the evolutionary context. In *Proceedings of the 12th annual conference comp on Genetic and evolutionary computation*, pages 2011–2014. ACM, 2010.
31. Adriana Lara, Gustavo Sanchez, Carlos A. Coello Coello, and Oliver Schütze. HCS: A New Local Search Strategy for Memetic Multi-Objective Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation*, 14(1):112–132, February 2010.

32. Adriana Lara, Oliver Schütze, and Carlos A. Coello Coello. New Challenges for Memetic Algorithms on Continuous Multi-objective Problems. In *GECCO 2010 Workshop on Theoretical Aspects of Evolutionary Multiobjective Optimization*, pages 1967–1970, Portland, Oregon USA, July 2010. ACM.
33. Erick Mejía. The directed search method for constrained multi-objective optimization problems. Master's thesis, CINVESTAV-IPN, México City, November 2010.
34. Erick Mejía and Oliver Schütze. A predictor corrector method for the computation of boundary points of a multi-objective optimization problem. In *Electrical Engineering Computing Science and Automatic Control (CCE), 2010 7th International Conference on*, pages 395–399. IEEE.
35. Pablo Moscato. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. *Caltech Concurrent Computation Program, C3P Report*, 826:1989, 1989.
36. J. Nocedal and S. Wright. *Numerical Optimization, Series in Operations Research and Financial Engineering*. Springer, New York, 2006.
37. Vilfredo Pareto. *Cours Déconomie Politique*. Lausanne, F. Rouge; Paris, Pichon, 1896.
38. S. Schäffler, R. Schultz, and K. Weinzierl. Stochastic method for the solution of unconstrained vector optimization problems. *Journal of Optimization Theory and Applications*, 114(1):209–222, 2002.
39. Oliver Schütze, Adriana Lara, and Carlos A. Coello Coello. The directed search method for unconstrained multi-objective optimization problems. In *EVOLVE 2011, A bridge between Probability, Set Oriented Numerics and Evolutionary Computation*.
40. Oliver Schütze, Adriana Lara, and Carlos A. Coello Coello. The directed search method for multi-objective optimization problems. Technical report, http://delta. cs. cinvestav. mx/˜schuetze/technical_reports/index. html, 2009.
41. Oliver Schütze, Adriana Lara, and Carlos A. Coello Coello. Evolutionary Continuation Methods for Optimization Problems. In *2009 Genetic and Evolutionary Computation Conference (GECCO'2009)*, pages 651–658, Montreal, Canada, July 8–12 2009. ACM Press. ISBN 978-1-60558-325-9.
42. Oliver Schütze, Adriana Lara, and Carlos A. Coello Coello. On the influence of the number of objectives on the hardness of a multiobjective optimization problem. *IEEE Transactions on Evolutionary Computation*, 15(4):444–455, August 2011.
43. Oliver Schütze, El-Ghazali Talbi, Carlos Coello Coello, Luis Vicente Santana-Quintero, and Gregorio Toscano Pulido. A Memetic PSO Algorithm for Scalar Optimization Problems. In *Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS 2007)*, pages 128–134, Honolulu, Hawaii, USA, April 2007. IEEE Press.
44. Pradyumn Shukla. Gradient Based Stochastic Mutation Operators in Evolutionary Multi-objective Optimization. In *Proceedings of the 8th international conference on Adaptive and Natural Computing Algorithms*, pages 58–66. Springer. Lecture Notes in Computer Science Vol. 4431, Warsaw, Poland, 2007.
45. Pradyumn Shukla. On Gradient Based Local Search Methods in Unconstrained Evolutionary Multi-objective Optimization. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*, pages 96–110, Matshushima, Japan, March 2007. Springer. Lecture Notes in Computer Science Vol. 4403.
46. Karthik Sindhya, Kalyanmoy Deb, and Kaisa Miettinen. A Local Search Based Evolutionary Multi-objective Optimization Approach for Fast and Accurate Convergence. In Günter Rudolph, Thomas Jansen, Simon Lucas, Carlo Poloni, and Nicola Beume, editors, *Parallel Problem Solving from Nature–PPSN X*, pages 815–824. Springer. Lecture Notes in Computer Science Vol. 5199, Dortmund, Germ., September 2008.
47. M. Vasile and F. Zuiani. A hybrid multiobjective optimization algorithm applied to space trajectory optimization. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, 2010.

48. M. Vasile and F. Zuiani. Multi-agent collaborative search: an agent-based memetic multi-objective optimization algorithm applied to space trajectory design. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 2011.
49. A.P. Wierzbicki. Reference point methods in vector optimization and decision support. Working Papers ir98017, International Institute for Applied Systems Analysis, April 1998.