HCS: A New Local Search Strategy for Memetic Multiobjective Evolutionary Algorithms

Adriana Lara, Gustavo Sanchez, Carlos A. Coello Coello, *Member, IEEE*, and Oliver Schütze

***Abstract*—In this paper, we propose and investigate a new local search strategy for multiobjective memetic algorithms. More precisely, we suggest a novel iterative search procedure, known as the *Hill Climber with Sidestep* (HCS), which is designed for the treatment of multiobjective optimization problems, and show further two possible ways to integrate the HCS into a given evolutionary strategy leading to new memetic (or hybrid) algorithms. The pecularity of the HCS is that it is intended to be capable both moving toward and along the (local) Pareto set depending on the distance of the current iterate toward this set. The local search procedure utilizes the geometry of the directional cones of such optimization problems and works with or without gradient information. Finally, we present some numerical results on some well-known benchmark problems, indicating the strength of the local search strategy as a standalone algorithm as well as its benefit when used within a MOEA. For the latter we use the state of the art algorithms Nondominated Sorting Genetic Algorithm-II and Strength Pareto Evolutionary Algorithm 2 as base MOEAs.**

***Index Terms*—Continuation, hill climber, memetic strategy, multiobjective optimization.**

I. Introduction

In a variety of applications in industry and finance, one is faced with the problem that several objectives have to be optimized concurrently leading to a *multiobjective optimization problem* (MOP). As a general example, two common goals in product design are certainly to maximize the quality of the product and to minimize its cost. Since these two goals are typically contradictory, it comes as no surprise that the solution set—the so-called *Pareto set*—of an MOP does not in general consist of one single solution but rather of an entire set of solutions (see Section II for a more detailed discussion).

For the computation of the Pareto set of a given MOP there exist several classes of algorithms. There exist, for instance, a variety of mathematical programming techniques such as scalarization methods (see e.g., [11], [17], [40] and references therein) or multiobjective continuation methods [24] which are in general very efficient in finding single solutions—the most prominent example is probably Newton's method which is used within continuation methods and which has local quadratic convergence [42]—or even entire sets of solutions but which may have trouble in finding the entire (global) Pareto set in certain cases. In contrast, there are global methods including multiobjective evolutionary algorithms (MOEAs) [10], [12] or subdivision techniques [15], [55] which accomplish the "global task" exceedingly but offer in turn (much) slower convergence rates compared to the algorithms mentioned above.

Another class of algorithms are the *memetic* (or hybrid) algorithms, i.e., algorithms that hybridize MOEAs with local search strategies (see Section II-B for an overview of existing methods). This is done in order to obtain an algorithm that offers on one hand the globality and robustness of the evolutionary approach, but on the other also an improved overall performance by the inclusion of well directed local search.

The scope of this paper is to contribute to the last category of algorithms. To be more precise, we propose a new point-wise local search prodecure, the *Hill Climber with Sidestep* (HCS), which is capable of moving both toward (using hill climber techniques) and along (sidestep) the Pareto set according to the distance of the current iterate to this set. In particular, the automatic switch of the movement represents a novelty that makes the operator universally applicable within any given MOEA. We present the HCS as local search procedure and demonstrate on two examples that it can be beneficial to integrating the HCS into a MOEA. According to the classification made in [58], the resulting algorithms nondominated sorting genetic algorithm (NSGA)-II-HCS and strength Pareto evolutionary algorithm (SPEA)2-HCS, which are based on NSGA-II and SPEA2, are exploitation-embedded hybrid methods.

The remainder of this paper is organized as follows. In Section II, we state some theoretical background and give an overview of the existing memetic MOEAs (MEMOEAs). In Section III, we introduce the underlying ideas of the HCS and propose two realizations: a gradient free version, and a version that exploits gradient information, both presented as standalone algorithms. In Section IV we address the integration of the HCS into a MOEA and propose two possible memetic strategies where NSGA-II and SPEA2 are used as base MOEAs. In Section V, we show some numerical results on both the HCS as a standalone algorithm as well as on the memetic strategies. Finally, some conclusions are drawn in Section VI.

II. Background

Here we briefly describe the background required for this paper; we introduce the notion of multiobjective optimization

Manuscript received November 06, 2008; revised April 13, 2009; accepted May 7, 2009. Current version published January 29, 2010. C. A. C. Coello acknowledges support from CONACyT Project No. 45683-Y.

A. Lara, C. A. Coello Coello, and O. Schütze are with the Departamento de Computación, Centro de Investigación y de Estudios Avanzados-Instituto Politécnico Nacional (CINVESTAV-IPN), México City, D.F. 07300, Mexico (e-mail: alara@computacion.cs.cinvestav.mx; ccoello@cs.cinvestav.mx; schuetze@cs.cinvestav.mx).

G. Sanchez is with the Department of Process and Systems, Simon Bolivar University, Caracas, Apartado 89000, Venezuela (e-mail: gsanchez@usb.ve).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2009.2024143

(MOO) and give an overview of existing memetic strategies for the numerical treatment of such problems.

A. MOO

In a variety of applications in industry and finance, a problem arises in which several objective functions have to be optimized concurrently leading to *multiobjective optimization problems* (MOPs). In the following, we consider continuous MOPs that are of the following form:

$$\min_{x\in Q}\{F(x)\}$$

where $Q \subset \mathbb{R}^n$ is the domain and the function F is defined as the vector of the objective functions

$$F : Q \to \mathbb{R}^k, \quad F(x) = (f_1(x), \dots, f_k(x))$$

and where each $f_i : Q \to \mathbb{R}$ is continuous. In this paper we will mainly consider the unconstrained case (i.e., $Q = \mathbb{R}^n$) but will give some possible modifications of the algorithms in case Q is defined by inequality constraints such as box constraints.

Central for the treatment of MOPs is the concept of the optimality of a point $x \in Q$ which is not analogous to the scalar objective case ($k = 1$). In the multiobjective case ($k > 1$), the concept of *dominance* is used which dates back over a century and was proposed first by Pareto [44].

Definition 1: (a) *Let $v, w \in \mathbb{R}^k$. Then the vector v is* less than *w ($v <_p w$), if $v_i < w_i$ for all $i \in \{1, \dots, k\}$. The relation $\le_p$ is defined analogously.*

(b) *A vector $y \in \mathbb{R}^n$ is* dominated *by a vector $x \in \mathbb{R}^n$ ($x \prec y$) with respect to* (1) *if $F(x) \le_p F(y)$ and $F(x) \neq F(y)$, else y is called non-dominated by x.*

(c) *A point $x \in Q$ is called* Pareto optimal *or a* Pareto point *if there is no $y \in Q$ which dominates x.*

In case all the objectives $f_i, i = 1, \dots, k$, of the MOP are differentiable, the following theorem of Kuhn and Tucker [38] states a necessary condition for Pareto optimality for unconstrained MOPs. For a more general formulation of the theorem we refer, e.g., to [40].

Theorem 1: Let x^* be a Pareto point of (*MOP*)*; then there exists a vector $\alpha \in \mathbb{R}^k$ with $\alpha_i \ge 0$, $i = 1, \dots, k$, and $\sum_{i=1}^k \alpha_i = 1$ such that*

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

The theorem claims that the vector of zeros can be written as a convex combination of the gradients of the objectives at every Pareto point. Obviously, (1) is not a sufficient condition for Pareto optimality. On the other hand, points satisfying (1) are certainly "Pareto candidates."

Definition 2: A point $x \in \mathbb{R}^n$ is called a Karush–Kuhn–Tucker point[1] (*KKT point*) *if there exist scalars $\alpha_1, \dots, \alpha_k \ge 0$ such that $\sum_{i=1}^k \alpha_i = 1$ and that* (1) *is satisfied.*

The set of all (globally) Pareto optimal solutions is called the *Pareto set*, denoted by P_Q. It has been shown that this set typically—i.e., under mild regularity assumptions—forms a $(k - 1)$-dimensional object [24]. The image of the Pareto set $F(P_Q)$ is called the *Pareto front*. Since we are involving local search strategies in our work we have to take also locally optimal points into consideration. In the following, let $\mathcal{P}$ be the set of local Pareto points. In case the MOP is differentiable, $\mathcal{P}$ can be considered as the set of KKT points.

Theorem 2 ([49]): Let (*MOP*) *be given and $q : \mathbb{R}^n \to \mathbb{R}^n$ be defined by*

$$q(x) = \sum_{i=1}^{k} \hat{\alpha}_i \nabla f_i(x) \tag{2}$$

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

$$\min_{\alpha\in\mathbb{R}^k} \left\{ \left\| \sum_{i=1}^{k} \alpha_i \nabla f_i(x) \right\|_2^2 ; \alpha_i \ge 0, i = 1, \dots, k, \sum_{i=1}^{k} \alpha_i = 1 \right\}. \tag{3}$$

Then either $q(x) = 0$ or $-q(x)$ is a descent direction for all objective functions $f_1, \dots, f_k$ in x.

The theorem 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 (3). In case $q(x) = 0$ the point x is a KKT point. Thus, a test for optimality has to be performed automatically when computing the descent direction for a given point $x \in Q$.

B. Memetic Strategies in MOO

Hybridization of MOEAs with local search algorithms has been investigated for more than 12 years, starting shortly after the first MOEAs were proposed [10], [36]. One of the first MEMOEAs for models on discrete domains was presented in [29], [30] as a "multiobjective genetic local search" approach. The authors proposed to use the local search method after classical variation operators are applied. A randomly drawn scalarizing function is used to assign fitness for parent selection.

Jaszkiewicz [32] proposed an algorithm called the Pareto memetic algorithm. This algorithm uses an unbounded "current set" of solutions and from this selects a small "temporary population" (TP) that comprises the best solutions with respect to a scalarizing function. Then TP is used to generate offspring by crossover. Jaszkiewicz suggests that scalarizing functions are particularly good at encouraging diversity than dominance ranking methods used in most MOEAs.

Another important MEMOEA, called memetic Pareto archived evolution strategy (M-PAES), was proposed in [35]. Unlike Ishibuchi's and Jaszkiewicz's approaches, M-PAES does not use scalarizing functions but employs instead a Pareto ranking-based selection coupled with a grid-type partition of the objective space. Two archives are used: one that maintains the global non-dominated solutions, and the other that is used as the comparison set for the local search phase.

In [41], the authors proposed a local search process with a generalized replacement rule. Ordinary two-replacement rules based on the dominance relation are usually employed in a local search for MOO. One is to replace a current solution with a solution which dominates it. The other is to replace the solution with a solution which is not dominated by it. The movable area with the first rule is very small when the number of objectives is large. On the other hand, it is too huge to move efficiently with the latter. The authors generalize these

[1] Named after the works of Karush [33] and Kuhn and Tucker [38].

extreme rules by counting the number of improved objectives for a given candidate.

Caponio and Neri [8] proposed the *cross dominant multiobjective memetic algorithm*, which consists of the NSGA-II combined with two local search engines: a multi-objective implementation of the Rosenbrock algorithm [47], which performs very small movements, and the Pareto domination multiobjective simulated annealing approach proposed in [61], which performs a more global exploration. The main idea of this approach is to use the mutual dominance between non-dominated solutions belonging to consecutive generations (this is called cross-dominance by the authors) as a parameter that indicates the degree of improvement achieved. Such value is applied with certain probability (based on a generalized Wigner semicircle distribution) to decide which of the two local search engines to apply. CDMOMA was found to have a similar or better performance than both NSGA-II and SPEA2 in several benchmark problems and in a real-world electrical engineering problem.

Soliman *et al.* [60] proposed a memetic version of a co-evolutionary multiobjective differential evolution (CMODE-MEM) approach, which evolves both a population of solutions and promising search directions. The fitness of a search direction is based on its capability to improve solutions. Local search is applied to a portion of the population after each generation. The performance of CMODE-MEM was assessed using several benchmark problems, and results were compared with respect to NSGA-II, NSDE [27], [28] and CMODE (without local search). The results indicated that the two versions of CMODE (with and without local search) were the best overall performers.

In [62]–[65], methods are presented which are hybrids of evolutionary search algorithms and multiagent strategies where the task of the agents is to perform the local search.

The continuous case—i.e., continuous objectives defined on a continuous domain—was explicitly first explored in [19], where a neighborhood search was applied to NSGA-II [13]. In their initial work, the authors applied the local search only after NSGA-II had ended. To do this, the authors applied a local search using a weighted sum of objectives. The weights were computed for each solution based on its location in the Pareto front such that the direction of improvement is roughly in the direction perpendicular to the Pareto front. Later works compare this approach with the same local search method being applied after every generation. Evidently, they found that the added computational workload impacted efficiency.

In [25] a gradient based local algorithm (sequential quadratic programming), was used in combination with NSGA-II and SPEA [72] to solve the 'ZDT benchmark suite' (named after the authors of [70], Zitzler, Deb, and Thiele). The authors stated that if there are no local Pareto fronts, the hybrid MOEA has faster convergence toward the true Pareto front than the original one, either in terms of the objective function evaluations or in terms of the CPU time consumed (since a gradient based algorithm is utilized, the sole usage of the number of function calls as a basis for a comparison can be misleading). Furthermore, they found that the hybridization technique does not decrease the solution diversity.

In [1], three different local search techniques were hybridized with multiobjective genetic algorithm (MOGA): simulated annealing, hill climbing, and tabu search. The three hybrid algorithms were applied to ZDT problems and compared to the standard MOGA considering the same number of function evaluations. An adaptive mechanism was proposed to determine the size of the neighborhood for each individual. The authors claim that the "MOGA-Hill climbing" was outperforming the standalone MOGA and the MOGA hybridized with the other local search techniques. They noted also that the process of fine-tuning the non-dominated individuals resulted in unwanted genetic drift and premature convergence: none of the hybrids was dedicated for distribution enhancement.

In [46], the authors proposed a hybrid technique that combines the robustness of MOGA-II [45] with the accuracy and speed of normal boundary intersection (NBI)-NLPQLP, which is an accurate and fast converging algorithm based on a classical gradient method. The methodology consists of starting with a preliminary robust MOGA-II run, and then isolating each single portion of the Pareto curve as an independent problem, each of which is treated with an independent accurate NBI-NLPQLP run.

In [68], the proposed local search process employs quadratic approximations for all objective functions. The samples gathered by the algorithm along the evolutionary process are used to fit these quadratic approximations around the point selected for local search. After that, a locally improved solution is estimated from the quadratic associated problem. The hybridization of the procedure is demonstrated with SPEA 2 [71].

A succesful hybrid approach was proposed in [26]. The authors proposed the algorithm MO-CMA-ES, which is a multiobjective CMA-ES [21] that combines the strategy parameter adaptation of evolutionary strategies with a multiobjective selection based on non-dominated sorting. The MO-CMA-ES is independent of the chosen coordinate system and its behavior does not change if the search space is translated, rotated, and/or rescaled. The authors claim that MO-CMA-ES significantly outperforms NSGA-II on all but one of the considered test problems: the NSGA-II is faster only on the ZDT4 problem where the optima form a regular axis-parallel grid, because NSGA-II heavily exploits this kind of separability.

In [67], a novel evolutionary algorithm (EA) for constrained optimization problems is presented: the so-called hybrid constrained optimization EA (HCOEA). The algorithm combines MOO with global and local search processes. In performing the global search, a niching genetic algorithm based on tournament selection is used. Meanwhile, the best infeasible individual replacement scheme is used as a local search operator for the purpose of guiding the population toward the feasible region of the search space. During the evolutionary process, the global search model effectively promotes high population diversity, and the local search model remarkably accelerates the convergence speed. HCOEA was tested on 13 benchmark functions, and the experimental results suggest that it is more robust and efficient than other state-of-the-art algorithms in terms of the selected performance metrics.

The use of gradient based hill climbing methods within NSGA-II has been proposed and studied in [57]. By this, the authors were able to accelerate the convergence of NSGA-II.

Finally, in [22], [52], [53], [55], hybrids can be found were heuristic methods are coupled with multiobjective continuation methods.

Concluding, it can be said that so far many authors have reported succesfull hybridizations of local search techniques with genetic algorithms. However, to the best of the authors' knowledge, there exist basically three crucial questions that remain open in the design of (single- or multiobjective) memetic strategies (e.g., [4], [23], [31], [37], [39], and [59]): Where shall a local search process be hybridized with a genetic algorithm? Which individuals should be fine-tuned and how much? And when shall the local refinement be applied?

In the following, we give a particular—but not all-embracing—response to these last questions.

III. HCS

In the following, we propose a novel iterative local search procedure, called the HCS, which is designed to be used within a memetic strategy. For sake of a better understanding, we present here the method as standalone algorithm. The integration of the HCS into MOEAs and the resulting modifications will be addressed in the next section.

Before we can come to the design of such a strategy, we have to ask ourselves what are the requirements for an iterative search procedure $\Phi : \mathbb{R}^n \to \mathbb{R}^n$ with

$$x_{l+1} = \Phi(x_l) \tag{4}$$

where $x_0 \in \mathbb{R}^n$ is a given initial solution and $\{x_i\}_{i \in \mathbb{N}_0}$ is the resulting sequence of iterates. Note that we are dealing with a point-wise iteration—i.e., input and output of Φ are a single points of the domain—and not with a population-based strategy. We are of the opinion that such a "wish list" on Φ for the treatment of MOPs includes the following tasks.

1) Φ should generate an improvement of the current iterate x_l if this one is not already "close" to the $\mathcal{P}$, i.e., a point x_{l+1} with $x_{l+1} \prec x_l$.
2) In case the current iterate x_l is already "close" to $\mathcal{P}$, a search *along* $\mathcal{P}$ would be desired.
3) The switch between the situations described in 1) and 2) should be done *automatically* according to the position of the current iterate x_l.
4) The process should work with or without gradient information (whether or not provided by the model).
5) The process should be capable of handling constraints of the MOP.

In 1) the "classical" task of a hill climber as known for single-objective optimization problems [16], [18], [34], [39], [43], [48] is described. 2) contains a pecularity of MOO, namely that there is—using the climbing metaphor—no single mountain top but rather an entire ridge of mountain tops which forms $\mathcal{P}$ (respectively a set of ridges in case $\mathcal{P}$ is diconnected). The generation of such a point x_{l+1} can be regarded as a "sidestep" relative to the current iterate x_l in the upward movement of the hill climber. Important for the efficiency of Φ within a memetic strategy is item 3), i.e., the capability to decide if case 1) or 2) is more appropriate.

In the following, we describe two variants of such a function Φ, which aims to fulfill the above wish list: one version of the HCS which is gradient free, and another version which involves gradient information.

A. HCS Without Using Gradient Information

First, we describe the HCS algorithm for the case in which no gradient information is available, since that seems to be more relevant for common real-world engineering problems, which is the main area of application for MOEAs. We concentrate here on the unconstrained case, and possible modifications of the algorithm for the treatment of MOPs with inequality constraints are given below.

The method we describe here is based on the geometry of MOO which has been studied in [7]. This paper gives a good insight into the structure of such problems by analyzing the geometry of the directional cones of candidate solutions at different stages of the optimization process: when a point x_0 is "far away" from any local Pareto optimal solution, the gradients' objectives are typically aligned and the descent cone is almost equal to the half-spaces associated with each objective. Therefore, for a randomly chosen search direction ν, there is a nearly 50% chance that this direction is a descent direction at x_0 (i.e., there exits an $h_0 \in \mathbb{R}_+$ such that $F(x_0 + h_0\nu) <_p F(x_0)$). If on the other hand a point x_0 is "close" to the Pareto set, the individual gradients are almost contradictory (compare also to the famous theorem of Kuhn and Tucker [38] which holds for points on $\mathcal{P}$), and thus the size of the descent cone is extremely narrow, resulting in a small probability for a randomly chosen vector to be a descent direction. The two scenarios are depicted in Fig. 1 for the bi-objective case. Hereby, $\{-,-\}$ and $\{+,+\}$ denote the descent and ascent cone, respectively. The symbol $\{-,+\}$ indicates that in this direction an improvement according to f_1 can be achieved while the values of f_2 will increase. To be more precise, if all objectives are differentiable, then the following equivalence holds for a search direction ν at a point x_0:

$$\nu \in \{-,+\} \Longleftrightarrow \langle \nabla f_1(x_0), \nu \rangle < 0 \text{ and } \langle \nabla f_2(x_0), \nu \rangle > 0 \tag{5}$$

where $\langle \cdot, \cdot \rangle$ denotes the standard scalar product. Analogous statements hold for $\{+,-\}$.

The gradient free HCS is constructed on the basis of these observations. Given a point $x_0 \in Q$, the next iterate x_1 is selected as follows: a further point $\tilde{x}_1$ is chosen randomly from a neighborhood of x_0, say $\tilde{x}_1 \in B(x_0, r)$ with

$$B(x_0, r) := \{x \in \mathbb{R}^n : x_{0,i} - r_i \le x_i \le x_{0,i} + r_i \;\forall i = 1, \ldots, n\} \tag{6}$$

where $r \in \mathbb{R}^n_+$ is a given (problem dependent) radius. If $\tilde{x}_1 \prec x_0$, then $\nu := \tilde{x}_1 - x_0$ is a descent direction[2] at x_0, and

[2]In the sense that there exists a $\bar{t} \in \mathbb{R}_+$ such that $f_i(x_0 + \bar{t}\nu) < f_i(x_0)$, $i = 1, \ldots, k$, but not in the "classical" sense, i.e., in case f_i is differentiable $\nabla f_i(x_0)^T \nu < 0$ is not guaranteed.

along it a "better" candidate can be searched, for example via line search methods (see below for one possible realization). If $x_0 \prec \tilde{x}_1$ the same procedure can be applied to the opposite direction (i.e., along $\nu := x_0 - \tilde{x}_1$) and starting with $\tilde{x}_1$. If x_0 is "far away" from any local solution, the chance is, by the above discussion, quite high that domination occurs, either $\tilde{x}_1 \prec x_0$ or $x_0 \prec \tilde{x}_1$. If x_0 and $\tilde{x}_1$ are mutually non-dominating, the process will be repeated with further candidates $\tilde{x}_2, \tilde{x}_3, \ldots, \in B(x_0, r)$. If only mutually non-dominated solutions $(\tilde{x}_i, x_0)$ are found within N_{nd} steps, this indicates, using the above observation, that the point x_0 is already near to the (local) Pareto set, and hence it is desirable to search along this set. This is because even if a descent direction will be available, further improvements will very likely be negligible, and, hence, it is advisable to seek for further regions of the Pareto set. To perform such a sidestep, it would be desirable to use the accumulated information obtained by the unsuccessful trials. Fundamental for the algorithm we present here is the fact that the "unsuccessful" search directions $\nu_{i,1} := \tilde{x}_i - x_0$ and $\nu_{i,2} := x_0 - \tilde{x}_i = -\nu_{i,1}$ are located in the diversity cones. Further, there exists the following relation of $\nu_{i,1}$ and $\nu_{i,2}$: if $\nu_{i,1}$ is, for example, in the cone $\{+,-\}$, then $\nu_{i,2}$ is the opposite cone $\{-,+\}$ which is a direct consequence of (5). This holds for bi-objective MOPs, the general k-objective case is analogous.

Based on these observations, we propose the following search directions. First we address the bi-objective case. If, for example, a search along $\{-,+\}$ after N_{nd} unsuccessful trials is sought, we propose to use the following one which uses the previous information:

$$\nu_{acc} = \frac{1}{N_{nd}} \sum_{i=1}^{N_{nd}} s_i \frac{\tilde{x}_i - x_0}{\|\tilde{x}_i - x_0\|} \tag{7}$$

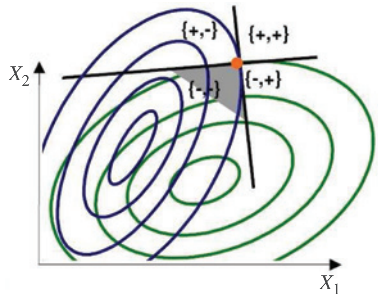
where

$$s_i = \begin{cases} 1, & \text{if } f_1(\tilde{x}_i) < f_1(x_0) \\ -1, & \text{else.} \end{cases} \tag{8}$$

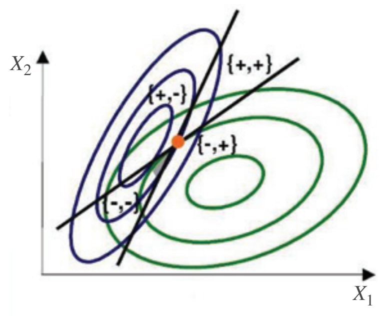
By construction, ν_{acc} is in $\{-,+\}$, and by the averaging of the search directions, we aim to obtain a direction which is "perpendicular" to the (small) descent cone. Note that in this case ν_{acc} is indeed a "sidestep" to the upward movement of the hill climbing process as desired, but this search direction does not neccessarily have to point along the Pareto set (see next section for a better guided search). A similar strategy for the search can be done for a general number k of objectives, however, leading to a larger variety for the search direction. For instance, for $k = 3$, there are six diversity cones which can be grouped by reflection as follows:

$$\begin{array}{lcl} \{+,-,-\} & \text{and} & \{-,+,+\} \\ \{+,-,+\} & \text{and} & \{-,+,-\} \\ \{+,+,-\} & \text{and} & \{-,-,+\}\,. \end{array} \tag{9}$$

That is, for $k = 3$ there are three different groups of cones in which search directions can be divided. (The sidesteps performed in Algorithm 6 of Section 4 are based on this idea.) For a general k, there are a total of $2^{k-1} - 1$ different groups making it less likely to find a perpendicular direction due to averaging within N_{nd} trials and within one of these cones. Alternatively to (7), one can, e.g., use the accumulated information by taking the average search direction over *all* search directions as follows:



(a)



(b)

Fig. 1. Descent cone (shaded) for an MOP with two parameters and two objectives during (a) initial and (b) final stages of convergence. The descent cone shrinks to zero during the final stages of convergence. The figure is taken from [4].

$$\nu_{acc} = \frac{1}{N_{nd}} \sum_{i=1}^{N_{nd}} \frac{\tilde{x}_i - x_0}{\|\tilde{x}_i - x_0\|}. \tag{10}$$

This direction has previously been proposed as a local guide for a multiobjective particle swarm algorithm in [6]. Note that this is a heuristic that does *not* guarantee that ν_{acc} indeed points to a diversity cone. In fact, it can happen that this vector points to the descent or ascent cone, though the probability for this is low for points x_0 "near" to a local solution due to the narrowness of these cones. However, in both cases Algorithm 1 acts like a classical hill climber—i.e., it searches for better points—which is still with in the scope of the procedure (though the improvements may not be significant due to the vicinity of the current iterate to $\mathcal{P}$).

A pseudocode of the HCS for the bi-objective case which uses the strategies described above and the sidestep heuristic (7) is given in Algorithm 1. In the following, we provide details for possible realizations of the line search and the handling of the constraints.

Sidestep Direction: The direction for the sidestep is determined by the value of i_0 (see line 5 and lines 15–20 of Algorithm 1). For simplicity, in Algorithm 1 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 of

Algorithm 1 HCS1 (without using gradient information for $k = 2$)

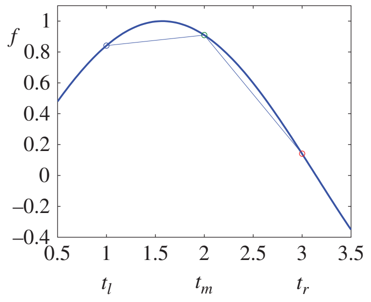
```
Require: starting point $x_0 \in Q$, radius $r \in \mathbb{R}^n_+$, number $N_{nd}$ of trials, MOP
    with $k = 2$
Ensure: sequence $\{x_l\}_{l \in \mathbb{N}}$ of candidate solutions
1: $a := (0, \ldots, 0) \in \mathbb{R}^n$
2: $nondom := 0$
3: $x_0^1 := x_0$
4: for $l = 1, 2, \ldots$ do
5:     set $x_l^1 := x_{l-1}^1$ and choose $x_l^2 \in B(x_l^1, r)$ at random
6:     choose $i_0 \in \{1, 2\}$ at random
7:     if $x_l^1 \prec x_l^2$ then
8:         $v_l := x_l^1 - x_l^2$
9:         compute $t_l \in \mathbb{R}_+$ and set $x_l^1 := x_l^2 + t_l v_l$.
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:        $v_l := x_l^2 - x_l^1$ and $x_l^1 := x_l^1 + t_l v_l$.
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 $x_l^1 := x_l^1 + \tilde{t}_l a$.
24:            $nondom := 0, \quad a := (0, \ldots, 0)$
25:        end if
26:    end if
27: end for
```

Algorithm 1) 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_{temp}. 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_{temp}, 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. 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 $\{+, -\}$ (this has in fact been chosen as the stopping criterion in Section V-A).

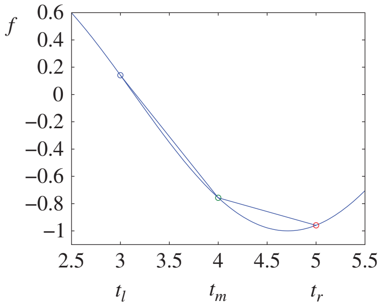
Computation of t_l: The situation is that we are given two points, say $x_0, x_1 \in \mathbb{R}^n$, such that $x_1 \prec x_0$. That is, there exists a subsequence $\{i_1, \ldots, i_l\} \subset \{1, \ldots, k\}$ with

$$f_{i_j}(x_1) < f_{i_j}(x_0), \quad j = 1, \ldots, l$$

and thus, $\nu := x_1 - x_0$ is a descent direction for all f_{i_j}'s at the point x_0. For this (single objective) case, there exist various strategies to perform the line search (see e.g., [16], [56]). One crucial problem is to find a good initial guess t^* for a suitable step size (which is, e.g., given by 1 when using Newton's method). In case t^* is not already sufficient, the step size can for instance be fine-tuned by backtracking methods ([16]). Since x_1 can be very close to x_0, the distance $\|x_1 - x_0\|$ cannot always serve as a good choice, and standard methods to obtain the initial guess do not apply. Instead, we propose the following heuristic to compute t^*. To capture the idea, we begin with the scalar case, i.e., we are given a function $f : \mathbb{R} \to \mathbb{R}$, and values $t_0, t_1 \in \mathbb{R}$ with $t_0 < t_1$ and $f(t_0) < f(t_1)$. We define $\Delta := t_1 - t_0$, $t_l := t_0$, $t_m := t_1$, and



(a)



(b)

Fig. 2. Term in (11) is false for t_l, t_m, and t_r in the top figure and true in the bottom figure. In the latter case, the quadratic polynomial is convex.

$t_r := t_0 + 2\Delta$ and check if

$$\frac{f(t_m) - f(t_l)}{t_m - t_l} < \frac{f(t_r) - f(t_m)}{t_r - t_m}. \tag{11}$$

If the above equation is true, we suggest to approximate f by a quadratic polynomial $p(t) = at^2 + bt + c$ (the values of a, b, and c can be derived explicitly by the interpolation conditions $p(t_l) = f(t_l)$, $p(t_m) = f(t_m)$, and $p(t_r) = f(t_r)$, see [16]). The reason for (11) is that if the term in (11) is true, then p is convex (see Fig. 2 for an example), and thus it is guaranteed that the extreme point of p, $t_p^* = -b/2a$, is a minimizer and takes its value in (t_0, ∞). Hence, t_p^* can be chosen as a guess for the minimizer of f. (This idea to approximate f locally by a quadratic polynomial was first proposed by Armijo [3].)

If (11) is false, quadratic approximation may not yield a useful result (in fact, in that case t_p^* may be negative), and we suggest to check condition (11) with the new data $t_l := t_m$, $t_m := t_r$, and $t_r := t_l + 4\Delta$ (i.e., doubling the step size for t_r). This process will be repeated until the boundary of the domain ∂Q is reached (in that case, take the maximal step size $t_{\max}$ as describe below) or (11) is true. The process will stop after a few iterations. If t_p^* is too large (i.e., if $f(t_p^*) > f(t_1)$), smaller step sizes can be found via backtracking.

In Algorithm 2, this idea is tranferred to the multiobjective case. Hereby, $f_{\nu,i}$ denotes the restriction of objective f_i to the line $x_0 + \mathbb{R}\nu$, i.e.,

$$f_{\nu,i}(t) = f_i(x_0 + t\nu) \tag{12}$$

and *quad_approx* the method to find the minimizer of the quadratic polynomial as described above.

Note that this step size control differs from the one presented in [51] since the initial guesses as described in [51] are

Algorithm 2 $t^* := hc\_step(x_0, x_1)$

```
Require: $x_0, x_1 \in \mathbb{R}^n$ with $x_1 \prec x_0$, maximal number of trials $N_{\max}$
Ensure: step size $t^*$ for the hill climber
 1: $I := \{i \in \{1, \dots, k\} : f_i(x_1) < f_i(x_0)\}$
 2: $\nu := x_1 - x_0$
 3: $\Delta := \|x_1 - x_0\|_2$
 4: $t_l := 0$, $t_m := \Delta$, $t_r := 2\Delta$
 5: for $j = 1, \dots, N_{\max}$ do
 6:     if $\exists i \in I$ : (11) is true for $t_l, t_m, t_r$ and $f_{\nu,i}$ then
 7:         for all $i \in I$ do
 8:             if (11) is true for $t_l, t_m, t_r$ and $f_{\nu,i}$ then
 9:                 $t_i^* := quad\_approx(t_l, t_m, t_r, f_{\nu,i})$
10:             else
11:                 $t_i^* := \infty$
12:             end if
13:         end for
14:         return $t^* := \min_{i=1,\dots,k} t_i^*$
15:     else
16:         $t_l := t_m$, $t_m := t_r$, $t_r := 2 * t_r$
17:     end if
18: end for
19: return $t^* := t_m$
```

restricted to the range $t^* \in (0, 2\|x_1 - x_0\|_2]$ which may be too small if x_1 is near to x_0.

Computation of $\tilde{t}_l$: We are given a point $x_0 \in \mathbb{R}^n$ and the search direction $a = \sum_{i=1}^{N_{nd}} s_i(\tilde{x}_i - x_0)$, $\|\tilde{x}_i - x_0\|$ [or alternatively direction (10)] with $\tilde{x}_i \in B(x_0, r)$, $i = 1, \dots, N_{nd}$, and such that $(x_0, \tilde{x}_i), i = 1, \dots, N_{nd}$ are mutually non-dominating. For this situation, we propose to proceed analogously to [50], where a step size strategy for multiobjective continuation methods is suggested: given a target value $\epsilon_y \in \mathbb{R}_+$, e.g., the minimal value which makes two solutions distinguishable from a practical point of view, the task is to compute a new candidate $x_{\text{new}} = x_0 + \tilde{t}a$ such that

$$\|F(x_0) - F(x_{\text{new}})\|_\infty \approx \epsilon_y. \tag{13}$$

In case F is Lipschitz continuous, there exists an $L \geq 0$ such that

$$\|F(x) - F(y)\| \leq L\|x - y\| \quad \forall x, y \in Q. \tag{14}$$

This constant can be estimated around x_0 by

$$L_{x_0} := \|DF(x_0)\|_\infty = \max_{i=1,\dots,k} \|\nabla f_i(x_0)\|_1$$

where $DF(x_0)$ denotes the Hessian of F at x_0 and $\nabla f_i(x_0)$ the gradient of the ith objective at x_0. In case the derivatives of F are not given (which is considered in this section), the accumulated information can be used to compute the estimation

$$\tilde{L}_{x_0} := \max_{i=1,\dots,N_{nd}} \frac{\|F(x_0) - F(\tilde{x}_i)\|_\infty}{\|x_0 - \tilde{x}_i\|_\infty}$$

since the $\tilde{x}_i$'s are near to x_0. Combining (13) and (14) and using the estimation L_{x_0} lead to the step size control

$$x_{\text{new}} = x_0 + \frac{\epsilon_y}{L_{x_0}} \frac{a}{\|a\|_\infty}. \tag{15}$$

Handling Constraints: In the course of the computation it can occur that iterates are generated which are not inside the feasible domain Q. That is, we are faced with the situation that $x_0 \in Q$ and $x_1 := x_0 + h_0\nu \notin Q$, where ν is the search direction. In that case, we propose to proceed analogously to the well-known bisection method for root finding in order to backtrack from the current iterate x_1 to the feasible set.

Algorithm 3 Backtracking to Feasible Region

```
Require: $x_0 \in Q$, $x_1 = x_0 + h_0\nu \notin Q$, $tol \in \mathbb{R}_+$
Ensure: $\tilde{x} \in \overline{x_0x_1} \cap Q$ with $\inf_{b\in\partial Q} \|b - \tilde{x}\| < tol$
 1: $in_0 := x_0$
 2: $out_0 := x_1$
 3: $i := 0$
 4: while $\|out_i - in_i\| \geq tol$ do
 5:     $m_i := in_i + \frac{1}{2}(out_i - in_i)$
 6:     if $m_i \in Q$ then
 7:         $in_{i+1} := m_i$
 8:         $out_{i+1} := out_i$
 9:     else
10:         $in_{i+1} := in_i$
11:         $out_{i+1} := m_i$
12:     end if
13:     $i := i + 1$
14: end while
15: return $\tilde{x} := in_i$
```

Algorithm 4 $h_{\max} := ComputeHmax(x_0, \nu, l, u)$

```
Require: feasible point $x_0 \in Q$, search direction $\nu \in \mathbb{R}^n\backslash\{0\}$, lower and
    upper bounds $l, u \in \mathbb{R}^n$
Ensure: maximal step size $h_{\max}$ such that $x_0 + h_{\max}\nu \in Q$
 1: for $i = 1, \dots, n$ do
 2:     if $v_i > 0$ then
 3:         $d_i := (u_i - x_i)/v_i$
 4:     else if $v_i < 0$ then
 5:         $d_i := -(x_i - l_i)/v_i$
 6:     else
 7:         $d_i := \infty$
 8:     end if
 9: end for
10: $h_{\max} := \min_{i=1,\dots,n} d_i$
```

Let $in_0 := x_0 \in Q$ and $out_0 := x_1 \notin Q$ and $m_0 := in_0 + 0.5(out_0 - in_0) = x_0 + (h_0/2)\nu$. If $m_0 \in Q$ set $in_1 := m_0$, else $out_1 := m_0$. Proceeding in an analogous way, one obtains a sequence $\{in_i\}_{i\in\mathbb{N}}$ of feasible points which converges linearly to the boundary ∂Q of the feasible set. One can, for example, stop this process with an $i_0 \in \mathbb{N}$ such that $\|out_{i_0} - in_{i_0}\|_\infty \leq tol$, obtaining a point in_{i_0} with maximal distance tol to ∂Q. See Algorithm 3 for one possible realization. Note that by this procedure no function evaluation has to be spent (though a feasibility test may also be of relevant numerical effort in some cases).

In case the domain Q is given by box constraints, i.e., if Q can be written as

$$Q = \left\{x \in \mathbb{R}^n : l_i \leq x_i \leq u_i,\ i = 1, \dots, n\right\} \tag{16}$$

where $l, u \in \mathbb{R}^n$ with $l \leq_p u$, the backtracking can be performed in one step, given a point $x_0 \in Q$ and a search direction ν, the maximal step size $h_{\max}$ such that $x_0 + h_{\max}\nu \in Q$ can be computed as shown in Algorithm 4.

Design Parameters: We agree that a realization of Algorithm 1 may include a variety of design parameters which may be difficult to tune and adapt to a particular problem. However, if the suggestions made in this paper are taken, merely the values for four design parameters have to be chosen (see Table I): the parameter r defines the neigborhood search of the procedure. Since this neighorbood search is used to find a search direction which is afterwards coupled with a

TABLE I
DESIGN PARAMETERS THAT ARE REQUIRED FOR THE REALIZATION OF THE GRADIENT FREE HCS ALGORITHM

Parameter	Description
r	Radius for neighborhood search (Algorithm 1)
N_{nd}	Number of trials for the hill climber before the sidestep is performed (Algorithm 1)
ϵ_y	Desired distance (in image space) for the sidestep (7)
tol	Tolerance value used for the backtracking in Algorithm 2

step size control, the value of r is not that important, but should be "small" to guarantee a local search. N_{nd} is the value that determines the number of directions which have to be averaged in order to choose the sidestep direction. In general, a larger value of N_{nd} leads to a "better" sidestep (in the sense that the search is performed orthogonal to the upward movement), but will in turn increase the cost of the search. We have experienced that a low value N_{nd}, say 5 to 10, already gives satisfactory results; the "accuracy" of the search does not seem to influence the performance of the HCS (unless the second derivatives of the objectives are available, see below). The value of ϵ_y is problem-dependent but can be given quite easily in a real world application [see discussion above (13)]. Finally, the tolerance *tol* has to be adjusted for constrained MOPs. The choice of this value is also problem-dependent and has to be chosen in every algorithm dealing with constraints.

B. HCS Using Gradient Information

In this section we discuss possible modifications that can be made to increase the performance of the HCS in case the MOP is sufficiently smooth. It will turn out that the resulting algorithm is more efficient (see Section V), but in turn, more information of the model is required.

Here we describe one possible realization of the HCS using the descent direction presented in Theorem 2 for the hill climber and some elements from multiobjective continuation for the sidestep:

Given a point $x \in \mathbb{R}^n$, the quadratic optimization problem (3) can be solved leading to the vector $\hat{\alpha}$. In case

$$\left\| \sum_{i=1}^{k} \hat{\alpha}_i \nabla f_i(x) \right\|_2^2 \geq \epsilon_{\mathcal{P}} \tag{17}$$

i.e., if the square of the norm of the weighted gradients is larger than a given threshold $\epsilon_{\mathcal{P}} \in \mathbb{R}_+$, the candidate solution x can be considered to be "away" from $\mathcal{P}$, and thus it makes sense to seek for a dominating solution. For this, the descent direction (2) can be taken together with a suitable step size control. For the latter, the step size control described above can be taken, or—probably better—a step size control which uses gradient information as, e.g., described in [16] or the one presented in [15]. If the value of the term in (17) is less than $\epsilon_{\mathcal{P}}$, this indicates that x is already in the vicinity of $\mathcal{P}$. In that case, one can lean elements from (multiobjective) continuation [2], [24] to perform a search along $\mathcal{P}$. To do this, we assume for simplicity that we are given a KKT point $\hat{x}$ and the according weight $\hat{\alpha}$ obtained by (3). Then the point

Algorithm 5 HCS2 (Using Gradient Information)

```
Require: starting point $x_0 \in Q$
Ensure: sequence $\{x_l\}_{l \in \mathbb{N}}$ of candidate solutions
 1: for $l = 0, 1, 2, \ldots$ do
 2:     compute the solution $\hat{\alpha}$ of (3) for $x_l$.
 3:     if $\| \sum_{i=1}^{k} \hat{\alpha}_i \nabla f_i(x_l) \|_2^2 \geq \epsilon_{\mathcal{P}}$ then
 4:         $\nu_l := -q(x_l)$
 5:         compute $t_l \in \mathbb{R}_+$ and set $x_{l+1} := x_l + t_l \nu_l$
 6:     else
 7:         compute $\tilde{F}'(\hat{x}, \hat{\alpha})^T = (Q_N, Q_K)U$ as in (19)
 8:         choose a column vector $\tilde{q} \in Q_K$ at random
 9:         compute $\tilde{t}_l \in \mathbb{R}_+$ and set $x_{l+1} := x_l + \tilde{t}_l \tilde{q}$.
10:     end if
11: end for
```

$(\hat{x}, \hat{\alpha}) \in \mathbb{R}^{n+k}$ is obviously 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:

$$\tilde{F}(x, \alpha) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i \nabla f_i(x) \\ \sum_{i=1}^{k} \alpha_i - 1 \end{pmatrix}. \tag{18}$$

In [24] 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

$$\tilde{F}'(\hat{x}, \hat{\alpha})^T = QU \in \mathbb{R}^{(n+k)\times(n+k)} \tag{19}$$

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 [24]—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 $\mathcal{P}$ and is thus well suited for a sidestep direction. The step size control can in this case taken exactly as proposed in (15) since the setting for that case was the same. In fact, since the search direction q_i is indeed pointing along $\mathcal{P}$, the results will be more accurate than for an averaged direction such as (7) or (10).

Algorithm 5 presents a procedure that is based on the above discussion. Note that this is one possible realization and that there exist certainly other possible ways leading, however, to similar results. For instance, alternatively to the descent direction used in Algorithm 5 the ones proposed in [17] and [5] can be taken. Further, the vicinity test (17) can be changed, though alternative conditions will most likely also be based on Theorem 2. Finally, the movement along $\mathcal{P}$ can be realized by predictor-corrector methods [2], [24] which consist, roughly speaking, of a repeated application of a predictor step obtained by a linearization of $\tilde{F}^{-1}(0)$ as in (19) and a corrector step which is done via a Gauss–Newton method.

Note that the HCS is proposed for the unconstrained case. While an extension to the constrained case for the hill climber is possible (see, e.g., [17] for possible modifications), this does not hold for the movement along the Pareto set (i.e., the sidestep). Though it is possible to extend system (18) by equality constraints (e.g., by introducing slack variables to transform

TABLE II
DESIGN PARAMETERS THAT ARE REQUIRED FOR THE REALIZATION OF THE HCS ALGORITHM WHICH INVOLVES GRADIENT INFORMATION

Parameter	Description
ϵ_y	Desired distance (in image space) for the sidestep (7)
tol	Tolerance value used for the backtracking in Algorithm 2
$\epsilon_{\mathcal{P}}$	Threshold for the vicinity test (17)

the inequality constraints into equality constraints), this could lead to effiency problems in the numerical treatment [24]. Hence, we restrict ourselves here to the unconstrained case.

As will be shown in Section V, the performance of the gradient based HCS in terms of convergence is better than its gradient free version, but this improvement does not come for free: for the descent direction all objectives' gradients have to be available (or approximated), and to perform the linearization of $\mathcal{P}$ even all second derivatives are required.

Design parameters: Analogous to the gradient free version of the HCS, the values of some design parameters have to be chosen for the realization of Algorithm 5. ϵ_y and tol are as discussed above, and N_{nd} and r are not needed due to the accuracy of the gradient based search. A new parameter, compared to the gradient free version of the HCS, is the threshold $\epsilon_{\mathcal{P}}$ for the vicinity test of a given candidate solution to $\mathcal{P}$. This value is certainly problem-dependent, but it can be made "small" due to the convergence properties of the hill climber (e.g., [17]).

IV. USE OF THE HCS WITHIN MOEAS

Here we address the integration of the HCS into a given MOEA. For this, we present some modifications required on the standalone version of the HCS to be able to be coupled efficiently with an evolutionary algorithm and discuss the cost of the procedure. Finally, we present two particular hybrids where NSGA-II and SPEA2 are used as base MOEAs.

A. Modifying the HCS

In Algorithms 1 and 5, the HCS is presented as standalone algorithm generating an infinite sequence of candidate solutions which is certainly not applicable when coupling it with a MOEA. To support the search of the latter algorithm, it is rather advisable to stop the iteration after a few iterations (denote this parameter by *maxiter*). In case the HCS finds only a sequence of dominating solutions (i.e., by the hill climber), merely the last dominating solution (denoted by x_d) has to be returned since the other intermetiate solutions are all dominated by x_d and are thus not important for the current population of the MOEA. In case the sidestep is performed, which indicates that the iterates are near to the (local) Pareto set, the iteratation can be stopped even before *maxiter* is reached. The second modification of HCS compared to the standalone version presented above that we suggest is to perform the sidestep in each diversity direction which has been found during the local search. This is due to the fact that the sidestep is the expensive part of the HCS (in terms of function calls, see also the discussion below), and hence all accumulated information should be exploited. Thus, the modified HCS will return in that case the dominating solution x_d (if not equal with the initial solution x_0) and further maximal $2^k - 2$ sidestep solutions in all diversity directions of x_d, depending on how many diversity directions of x_p have been found within the N_{nd} "unsuccessful" trials.

Algorithm 6 shows such a modification of the HCS1 for $k = 3$. Thereby

$$C(x, s_1, s_2, s_3) \tag{20}$$

where $s_i \in \{+, -\}, i = 1, 2, 3$, denotes the diversity cone at a point x. For instance, it is

$$\begin{aligned} y \in C(x, +, +, -) :\Leftrightarrow \{ & f_1(y) > f_1(x) \quad \text{and} \\ & f_2(y) > f_2(x) \quad \text{and} \quad f_3(y) < f_3(x)\}. \end{aligned} \tag{21}$$

The algorithm requires the starting point x_0 and returns the set X_{new} which can consist of one candidate solution (i.e., the result of the hill climber x_d) up to seven candidate solutions [x_d plus candidates in all the six diversity directions (9) of x_d].

For HCS2, the modifications described above are much easier to handle: if the sidestep is performed [i.e., if (17) is false] the sidestep solutions can be chosen as

$$x_+^i := x_d + h_i q_i, \quad x_-^i := x_d - h_i q_i \tag{22}$$

for all column vectors q_i of Q_K, which leads to $2k - 2$ new candidate solutions.

B. Cost of the HCS

Crucial for the efficient usage of the HCS within a MOEA is the knowledge of its cost. Here we measure the cost of one step of the modified HCS as described above (i.e., for *maxiter* = 1). Unfortunately, the different algorithms use different information (mainly different gradient information) of the model. For sake of comparison, we measure the cost of the HCS in terms of required function calls (to be more precise, we measure the running time for a function call and neglect the memory requirement). That is, to measure HCS2 we have to find an equivalent in terms of function calls for the computation or approximation of the derivative $\nabla f(x)$ and the second derivative $\nabla^2 f(x)$ of a function $f : \mathbb{R}^n \to \mathbb{R}$ at a point x. If for instance automatic differentiation (AD) is used to compute the derivatives, we can estimate 5 function calls for the derivative call and $4 + 6n$ function call for the second derivative [20]. These values change when using finite differences (FD). If for instance the forward difference quotient

$$\frac{\partial f}{\partial x_i}(x) \approx \frac{f(x_1, \ldots, x_i + \delta_i, \ldots, x_n) - f(x_1, \ldots, x_n)}{\delta_i}, \quad i \in \{1, \ldots, n\} \tag{23}$$

where $\delta_i \in \mathbb{R}_+$ is a small value, is used to estimate the gradient, apparently n function calls are required. The central difference quotient leads to more accurate approximations, but does in turn require $2n$ function calls ([20]). A forward difference quotient approximation of the second derivative requires a total of n^2 function calls (and $2n^2$ or $4n^2$ function calls when using the central difference quotient, depending

Algorithm 6 HCS1 (for use within MOEAs for $k = 3$)

```
Require: maximal number of iterations $maxiter$, rest as in Algorithm 1
Ensure: set of candidate solutions $X_{\text{new}}$
  $L_1 := 0,\ L_2 := 0,\ L_3 := 0$
  $a_1 := 0 \in \mathbb{R}^n,\ a_2 := 0 \in \mathbb{R}^n,\ a_3 := 0 \in \mathbb{R}^n$
  $no\_a_1 := 0,\ no\_a_2 := 0,\ no\_a_3 := 0$
  $nondom := 0$
  $x_{1,0} := x_0$
  for $i = 1, 2, \ldots, maxiter$ do
      for $j = 1, 2, \ldots, N_{nd}$ do
          choose $x_2 \in B(x_{1,i-1}, r)$ at random
          if $x_{1,i-1} \prec x_2$ then
              compute $t \in \mathbb{R}_+$ as in Algorithm 1 (l. 6–10), set $x_{1,i} := x_2 + t(x_{1,i-1} - x_2)$.
              $nondom := 0, \quad a_1 = a_2 = a_3 = 0$
              continue
          else if $x_2 \prec x_{1,i-1}$ then
              comp. $t \in \mathbb{R}_+$ as in Algorithm 1 (l. 11–13), set $x_{1,i} := x_{1,i-1} + t(x_2 - x_{1,i-1})$.
              $nondom := 0, \quad a_1 = a_2 = a_3 = 0$
              continue
          else
              $nondom := nondom + 1$
              if $x_2 \in C(x_{1,i-1}, -, -, +)$ then
                  $a_1 := a_1 + (x_2 - x_{1,i-1})/\|x_2 - x_{1,i-1}\|_\infty$
                  $no\_a_1 := no\_a_1 + 1$
                  $L_1 := \max(L_1, \|F(x_2) - F(x_{1,i-1})\|_\infty/\|x_2 - x_{1,i-1}\|_\infty)$
              end if
              if $x_2 \in C(x_{1,i-1}, +, +, -)$ then
                  $a_1 := a_1 + (x_{1,i-1} - x_2)/\|x_{1,i-1} - x_2\|_\infty$
                  $no\_a_1 := no\_a_1 + 1$
                  $L_1 := \max(L_1, \|F(x_{1,i-1}) - F(x_2)\|_\infty/\|x_{1,i-1} - x_2\|_\infty)$
              end if
              if $x_2 \in C(x_{1,i-1}, -, +, -)$ then
                  update $a_2, no\_a_1$, and $L_2$ analogous to lines 19–22.
              end if
              if $x_2 \in C(x_{1,i-1}, +, -, +)$ then
                  update $a_2, no\_a_2$, and $L_2$ analogous to lines 24–27.
              end if
              if $x_2 \in C(x_{1,i-1}, +, -, -)$ then
                  update $a_3, no\_a_3$, and $L_3$ analogous to lines 19–22.
              end if
              if $x_2 \in C(x_{1,i-1}, -, +, +)$ then
                  update $a_3, no\_a_3$, and $L_3$ analogous to lines 24–27.
              end if
          end if
      end for
      $X_{\text{new}} := \{x_{1,i}\}$                                   ▷ perform sidesteps and return
      if $no\_a_1 > 0$ then
          $v_1 := a_1/\|a_1\|_\infty, \quad h_1 := \epsilon_y/L_1$
          $x_+^{(1)} := x_{i,N_{nd}} + h_1 v_1, \quad x_-^{(1)} := x_{i,N_{nd}} - h_1 v_1$
          $X_{\text{new}} := X_{\text{new}} \cup \{x_+^{(1)}, x_-^{(1)}\}$
      end if
      if $no\_a_2 > 0$ then
          compute $x_+^{(2)}, x_-^{(2)}$ analogous to lines 44–47, set $X_{\text{new}} := X_{\text{new}} \cup \left\{x_+^{(2)}, x_-^{(2)}\right\}$
      end if
      if $no\_a_3 > 0$ then
          compute $x_+^{(3)}, x_-^{(3)}$ analogous to lines 44–47, set $X_{\text{new}} := X_{\text{new}} \cup \left\{x_+^{(3)}, x_-^{(3)}\right\}$
      end if
      return
  end for
  $X_{\text{new}} := \{x_{1,maxiter}\}$                                   ▷ return dominating solution
```

on how the rule is applied). Finally, we have to estimate the number of function calls required for a line search for the hill climber. Here we take the value of 3 obtained by our observations.

A call of HCS1 requires at least four function calls: one for the local search around x_0. If the new candidate solution is either dominating or dominated by x_0—which is very likely in the early stage of the optimization process—the next point is found via line search resulting in four function calls due to our assumptions. When a sidestep, which is the most expensive event of the HCS, is performed, this means that first N_{nd} trials have been made around x_0, and then candidates in maximal $2^k - 2$ directions are computed [for each one function call is required, see (15)] leading to a total of $N_{nd} + 2^k - 2$ function calls.

The HCS2 needs for the realization of the hill climber the gradients of all k objectives, the solution of QOP (which we do not count here since k is typically low, and thus, the quadratic problem is easy to solve with standard techniques), and one line search. This makes $5k + 3$ function calls when using AD

TABLE III
COST OF ONE STEP OF THE HCS MEASURED IN FUNCTION CALLS. TO CONVERT THE DERIVATIVE CALLS IN HCS2 INTO FUNCTION CALLS WE HAVE USED VALUES BASED ON AD AND FD

Method	No. of function calls required
HCS1	from 4 to $N_{nd} + 2^k - 2$
HC2 (AD)	$5k + 3$
HC2 (FD)	$kn + 3$
HCS2 (AD)	from $5k + 3$ to $6n + 7k + 2$
HCS2 (FD)	from $kn + 3$ to $n^2 + k(n + 2) - 2$

TABLE IV
NUMERICAL VALUES FOR THE COST OF THE HCS ALGORITHMS FOR THE SETTINGS (A) $n_1 = 10$, $k = 3$, $N_{nd} = 3$ AND (B) $n_2 = 10$, $k = 3$, $N_{nd} = 3$. SEE TABLE III FOR DETAILS

Method	No. of function calls required for $n_1 = 10$, $k = 3$, $N_{nd} = 3$	No. of function calls required for $n_2 = 30$, $k = 3$, $N_{nd} = 3$
HCS1	From 4 to 9	From 4 to 9
HC2 (AD)	18	18
HC2 (FD)	33	93
HCS2 (AD)	From 18 to 83	From 18 to 203
HCS2 (FD)	From 33 to 138	From 93 to 996

and $kn+3$ function calls when using FD (here we assume the forward difference method). For a sidestep, k gradients and the second derivative of $\sum_{i=1}^{k} \alpha_i f_i(x_d)$ have to be computed, and further $2k - 2$ sidestep candidates are produced. This leads to $6n + 7k + 2$ function calls when using AD and to $n^2 + k(n + 2) - 2$ function calls when using FD.

Table III summarizes the cost of the different algorithms using the different conversion rules. Here, HC2 denotes the hill climber as presented in Algorithm 5 but without the sidestep operator (i.e., for $\epsilon_{\mathcal{P}} = 0$). Table IV gives the numerical values for $k = 3$, $N_{nd} = 3$, $n_1 = 10$, and $n_2 = 30$. It is obvious that FD should only be used for models with moderate dimensional parameter space since else the cost of the HCS2 will get tremendous. On the other hand, note that the cost of HCS1 is independent of n and thus relatively inexpensive, in particular in higher dimensions.

C. Integration into MOEAs

The questions which remains open is how to integrate the HCS into a given MOEA in order to obtain an efficient memetic strategy. Here we make first steps to answer this problem and propose hybrids with the state-of-the-art MOEAs NSGA-II and SPEA2. The numerical results in the next section show that the combination is advantageous, however, we think that more effort has to be made to obtain a universal and self-adaptive memetic algorithm which is beyond the scope of this paper.

The modified HCS can be written in the shorthand form as

$$P_{HCS} = HCS\,(x_0) \tag{24}$$

where x_0 is a given point (e.g., coming from the current population of the MOEA) and P_{HCS} is the output set. Given a probability p_{HCS} for the application of the procedure on

Algorithm 7 NSGA-II-HCS

```
1: procedure NSGA-II-HCS($N$, $G$, $p_{HCS}$, $s$)
2:     Generate Random Population P (size $N$).
3:     Evaluate Objective Values.
4:     Fast Non-Dominated Sort
5:     Crowding Distance Assignment
6:     Generate Child Population $P_{offs}$
7:     for $i := 1, \ldots, G$ do
8:         Using $P := P \cup P_{offs}$:
9:         if mod(i, s)==0 then
10:            LocalSearch($p_{HCS}$)
11:        end if
12:        Fast Non-Dominated Sort
13:        Crowding Distance Assignment
14:        Generate Child Population $P_{offs}$
15:    end for
16: end procedure

17: procedure LOCALSEARCH($p_{HCS}$)
18:    for all $a \in P$ do
19:        if $\nexists b \in P$ such that $b \prec a$ then
20:            $A_a = HCS(\{a\}, p_{HCS})$
21:            $P := P \cup A_a$
22:        end if
23:    end for
24: end procedure
```

Algorithm 8 SPEA2–HCS

```
Require: $N$, $\bar{N}$, $N_{maxiter}$
Ensure: $A$ (nondominated set)
1: Generate an initial population $P_0 \subset Q$ and create $A_0 := \emptyset$, $\bar{P}_0 := \emptyset$.
2: for $k = 0, 1, \ldots, N_{maxiter}$ do
3:     Calculate fitness values of individuals in $P_k$ and $\overline{P}_k$
4:     Copy all non-dominated individuals in $P_k$ and $\overline{P}_k$ to $\overline{P}_{k+1}$
5:     If size of $\overline{P}_{k+1}$ exceeds $\bar{N}$ then reduce $\overline{P}_{k+1}$ by means of the
           truncation operator, otherwise if size of $\overline{P}_{k+1}$ is less than $\bar{N}$ then
           fill $\overline{P}_{k+1}$ with dominated individuals in $P_k$ and $\overline{P}_k$
6:     Set $A_{k+1}$ := nondominated solutions of $\overline{P}_{k+1}$
7:     Perform binary tournament selection in $\overline{P}_{k+1}$ to fill the mating pool
8:     Apply crossover, mutation and the local search operator (HCS) to the
           mating pool. Denote the resulting population by $P_{k+1}$
9: end for
```

an individual of a population, the operator can be defined set-wise as

$$P_{HCS} = HCS\,(P, p_{HCS}) \tag{25}$$

where P denotes a given population. By doing so, the HCS can be interpreted as a particular mutation operator, and, thus can in principle be integrated into any given MOEA with little effort. However, this should be handled with care since the efficiency of the resulting hybrid depends (among others) on 1) which elements of the population the HCS is applied to, and 2) the balance of the genetic search and the HCS (see also Section II-B). As an example for 1) we have observed that if the HCS is merely applied on elements of the external archive in a combination with SPEA2, that this "elitism approach" has a negative effect on the diversity of the population, at least in early stages of the search. Even the application of the sidestep cannot compensate this effect, since it is applied on a few, possibly closely located solutions. Problem 2) is another typical problem when designing memetic strategies (probably first reported in [31] in the context of MOO), and in particular in our setting due to the relatively high cost of the HCS compared to classical mutation operators.

Most important for the effect and the cost of the HCS are the parameters *maxiter* and N_{nd} (for HCS1). In general,

TABLE V
MOPs UNDER INVESTIGATION IN THIS WORK. HEREBY, $\tilde{k} = n - k + 1$

CONV1
$f_1(x) = (x_1 - 1)^4 + \sum_{j=2}^{n}(x_j - 1)^2$
$f_2(x) = \sum_{j=1}^{n}(x_j + 1)^2$
CONV2
$f_i(x) = \sum_{\substack{j=1 \\ j \neq i}}^{n} (x_j - a_j)^2 + (x_i - a_i)^4, \; i = 1, 2, 3$
$a_1 = (1, \ldots, 1) \in \mathbb{R}^n$
$a_2 = (-1, \ldots, -1) \in \mathbb{R}^n$
$a_3 = (1, -1, 1, -1 \ldots) \in \mathbb{R}^n$
ZDT4
$f_1(x) = x_1$
$f_2(x) = g(x)(1 - \sqrt{f_1, g(x)})$
$g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}\left(x_i^2 - 10cos(4\pi x_i)\right)$
$0 \leq x_1 \leq 1, \quad -5 \leq x_i \leq 5, \; i = 2, \ldots, n$
DTLZ1
$f_1(x) = \frac{1}{2} x_1 x_2 \ldots x_{k-1}(1 + g(x))$
$f_2(x) = \frac{1}{2} x_1 x_2 \ldots (1 - x_{k-1})(1 + g(x))$
$\vdots$
$f_{k-1}(x) = \frac{1}{2} x_1 (1 - x_2)(1 + g(x))$
$f_k(x) = \frac{1}{2}(1 - x_1)(1 + g(x))$
$g(x) = 100\left[\tilde{k} + \sum_{i=k}^{n}(x_i - \frac{1}{2})^2 - \cos\left(20\pi\left(x_i - \frac{1}{2}\right)\right)\right]$
$0 \leq x_i \leq 1, \; i = 1, \ldots, n$
DTLZ2
$f_1(x) = \cos\left(\frac{x_1\pi}{2}\right)\cos\left(\frac{x_2\pi}{2}\right)\ldots\cos\left(\frac{x_{k-1}\pi}{2}\right)(1 + g(x))$
$f_2(x) = \cos\left(\frac{x_1\pi}{2}\right)\cos\left(\frac{x_2\pi}{2}\right)\ldots\sin\left(\frac{x_{k-1}\pi}{2}\right)(1 + g(x))$
$\vdots$
$f_{k-1}(x) = \cos\left(\frac{x_1\pi}{2}\right)\sin\left(\frac{x_2\pi}{2}\right)(1 + g(x))$
$f_k(x) = \sin\left(\frac{x_1\pi}{2}\right)(1 + g(x))$
$g(x) = \sum_{i=k}^{n}\left(x_i - \frac{1}{2}\right)^2$
$0 \leq x_i \leq 1, \; i = 1, \ldots, n$
DTLZ3
$f_1(x) = \cos\left(\frac{x_1\pi}{2}\right)\cos\left(\frac{x_2\pi}{2}\right)\ldots\cos\left(\frac{x_{k-1}\pi}{2}\right)(1 + g(x))$
$f_2(x) = \cos\left(\frac{x_1\pi}{2}\right)\cos\left(\frac{x_2\pi}{2}\right)\ldots\sin\left(\frac{x_{k-1}\pi}{2}\right)(1 + g(x))$
$f_{k-1}(x) = \cos\left(\frac{x_1\pi}{2}\right)\sin\left(\frac{x_2\pi}{2}\right)(1 + g(x))$
$f_k(x) = \sin\left(\frac{x_1\pi}{2}\right)(1 + g(x))$
$g(x) = 100\left[\tilde{k} + \sum_{i=k}^{n}\left(x_i - \frac{1}{2}\right)^2 - \cos\left(\alpha\pi\left(x_i - \frac{1}{2}\right)\right)\right]$
$\alpha = 20$
$0 \leq x_i \leq 1, \; i = 1, \ldots, n$

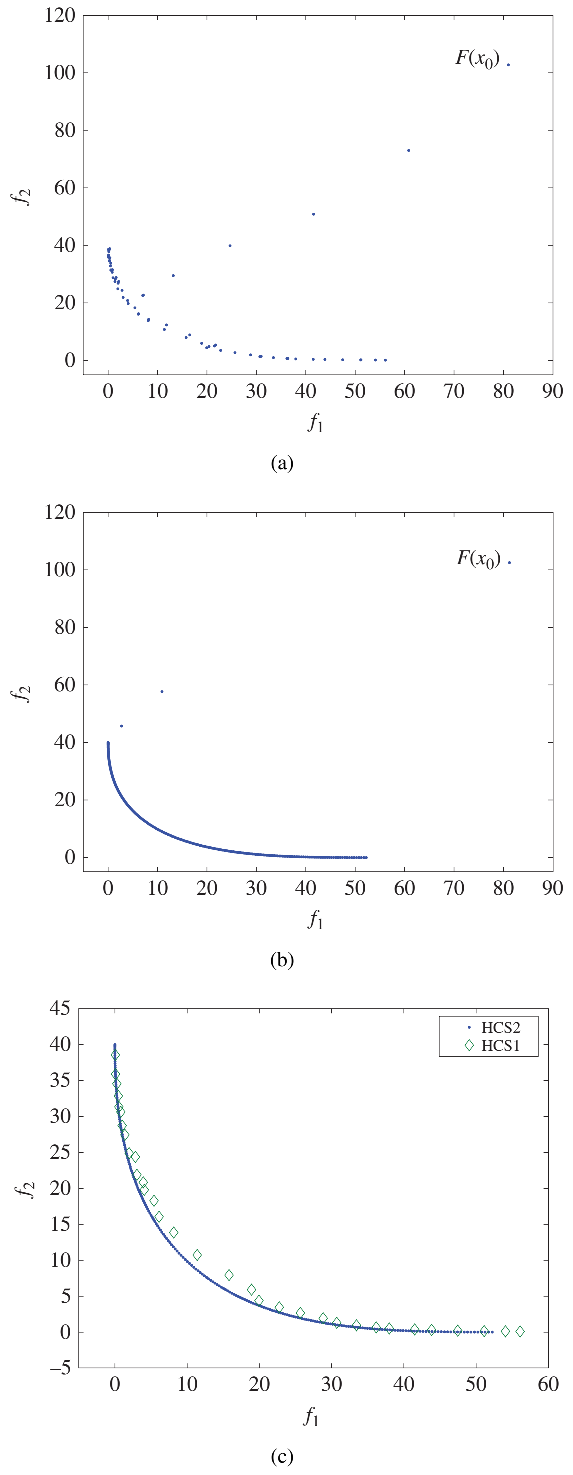


Fig. 3. Numerical result of HCS for MOP CONV1 with $Q = [-5, 5]^{10}$ in objective space (unconstrained case). (a) Solution HCS1, (b) solution HCS2, and (c) comparison non-dominated fronts.

it can be said that if both values of *maxiter* and N_{nd} are high, the local improvement of a point x_0 will be nearly optimal (i.e., the elements of P_{HCS} will be near to local solutions). This can be advantageous for unimodal models but can in turn reduce the efficiency of the entire search algorithm for multimodal models due to the high cost of the HCS and the relatively high chance that the search gets stuck in a local (and not global) solution. If the values of *maxiter* and N_{nd} are low, the local improvements in one application of HCS will typically be suboptimal. However, the choice of low values offers on the other hand two advantages. First, the HCS spends less function calls for unpromising starting points. That is apparently also the case for promising starting points, but we have observed that it is advantageous to repeat the local search more often instead to spend the function calls for single solutions (future populations contain points which are at least as good as the point x_0 from the current population). The second advantage

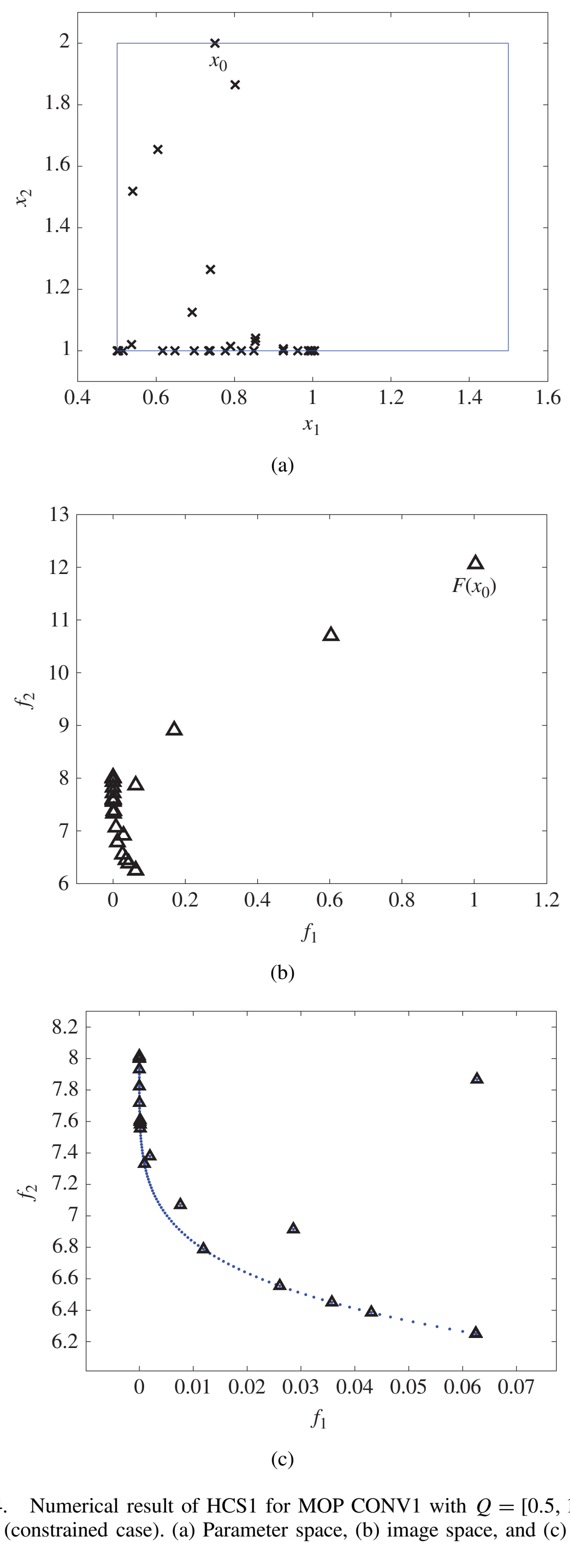


Fig. 4. Numerical result of HCS1 for MOP CONV1 with $Q = [0.5, 1.5] \times [1, 2]$ (constrained case). (a) Parameter space, (b) image space, and (c) zoom image space and Pareto front.

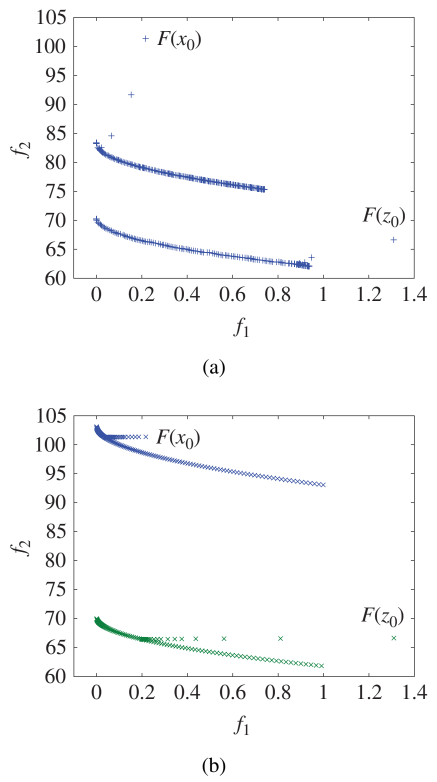


Fig. 5. Numerical result of HCS1 and HCS2 for MOP ZDT4 in objective space for two initial solutions x_0 and z_0. (a) Result HCS1 and (b) result HCS2.

TABLE VI
PARAMETER VALUES USED FOR SPEA2 AND NSGA-II AND THE MEMETIC STRATEGIES SPEA2-HCS AND NSGA-II-HCS ON THE CONVEX PROBLEMS

	SPEA2-HCS		NSGA-II-HCS	
Parameters	Unbounded	Bounded	Unbounded	Bounded
N_{pop}	100	100	100	100
N_a	100	100	—	—
η_c	—	—	15	15
η_m	—	—	20	20
p_c	0.9	0.9	0.9	0.9
p_m	0.1	0.1	1/30	1/30
p_{HCS1}	0.2	0.2	—	—
s_{HCS1}	5	5	10	10
p_{HCS2}	0.1	0.1	—	—
s_{HCS2}	10	10	10	10
ε_y	5	2	5	5
$\varepsilon_{\mathcal{P}}$	0.0001	0.0001	0.0001	0.0001
r	0.1	0.02	0.1	0.1
maxiter	10	10	10	10
N_{nd}	5	5	3	3
tol	10^{-4}	10^{-4}	10^{-4}	10^{-4}

is that the population is not disturbed by drastic improvements of single solutions which may cause trouble in elitist strategies ([35]).

The next question is the choice of the probability p_{HCS} to apply the HCS. Due to the cost of the HCS, a low value seems to be advisable, which also coincides with our observations. Further, we suggest not to apply the HCS in every generation in order not to disturb the efficient but highly sensitive interplay of the different operators of the MOEA (as e.g., done in [69]).

To summarize, we suggest low values for the parameters *maxiter* and N_{nd} which influence efficiency and cost of one application of the HCS, and a low value for the probability p_{HCS} of its application which influences the overall cost of the

TABLE VII
NUMERICAL RESULTS OF SPEA2 AND THE MEMETIC STRATEGIES SPEA2-HCS ON THE CONVEX PROBLEMS USING DIMENSION $n = 30$ AND PERFORMING 10 000 FUNCTION CALLS (MEAN VALUES AND STANDARD DEVIATIONS IN BRACKETS). STATISTICS WERE GATHERED FROM 30 INDEPENDENT RUNS

		Indicators	
Problems		GD	IGD
CONV1-U	SPEA2(A)	3.5762(0.9190)	2.5469(0.3460)
	SPEA2-HCS1(B)	1.9399(0.5721)	2.2846(0.5483)
	SPEA2-HCS2(C)	0.1139(0.0505)	1.8377(0.9405)
	SPEA2-HC2(D)	0.1483(0.0463)	1.8792(0.9371)
CONV1-C	SPEA2(A)	6.8446(2.7245)	1.2491(0.1268)
	SPEA2-HCS1(B)	6.3508(1.9602)	1.1985(0.1225)
	SPEA2-HC2(C)	0.2800(1.1503)	0.4268(0.0069)
CONV2-U	SPEA2(A)	3.5762(0.9190)	2.5469(0.3460)
	SPEA2-HCS1(B)	1.9399(0.5721)	2.2846(0.5483)
	SPEA2-HCS2(C)	0.1139(0.0505)	1.8377(0.9405)
	SPEA2-HC2(D)	0.1483(0.0463)	1.8792(0.9371)
CONV2-C	SPEA2(A)	1.7887(0.4642)	0.4882(0.1086)
	SPEA2-HCS1(B)	1.6342(0.7454)	0.4341(0.1303)
	SPEA2-HC2(C)	0.4902(0.1789)	0.2774(0.1440)

Set Coverage CONV1-U					
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$	$D \prec A$	$A \prec D$
0.7834(0.2726)	0.0984(0.1929)	0.9742(0.0755)	0(0)	0.9770(0.0753)	0(0)

Set Coverage CONV1-C			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.4881(0.4119)	0.2932(0.3604)	1(0)	0(0)

Set Coverage CONV2-U					
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$	$D \prec A$	$A \prec D$
0.5602(0.2900)	0.1223(0.1405)	0.8963(0.2118)	0(0)	0.9893(0.0324)	0(0)

Set Coverage CONV2-C			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.6426(0.3622)	0.1796(0.2752)	1(0)	0(0)

local search. See next section for particular choices of these values.

In the following, we propose two particular combinations where we use NSGA-II and SPEA2 as base MOEAs.

1) NSGA-II-HCS: As discussed above, crucial are the questions when and to which elements the local search has to be applied within a given MOEA. For NSGA-II, we suggest to perform the local search (i.e., HCS) only on the best individuals of a given generation. This is made in order to find leader individuals to pull the entire population to better solutions during the search. This exclusive search can be done since the diversity of the best (i.e., non-dominated) solutions is typically quite high, also in early stages of the search. Thus, it is likely to generate well-spread leader individuals from the beginning of the search, which helps to pull the population to the entire Pareto set.

Algorithm 7 shows one which combines NSGA-II with HCS. Here, the procedures "Fast Non-Dominated Sort," "Crowding Distance Assignment" and "Generate Child Population" are well known as parts of the NSGA-II; a thorough discussion can be found in [13].

Algorithm 7 applies the local search each s generation after reproduction. The local search is applied only to non-dominated individuals, and, due to the cost of the procedure, is performed with a certain (low) probability. After having computed the improved solutions of local search, the regular operations of ranking and crowding are used as in NSGA-II.

Contrary to [51], where the local search has been applied after 75% of a given budget B of function calls have been spent, we have observed that it is advantageous to apply the HCS in all stages of the search to pull the population permanently to the Pareto set. In fact, we propose here that the local search be evenly distributed over the run of the algorithm. This guideline and the choice to take only non-dominated solutions as starting points for the HCS has an implication on the rule to choose p_{HCS}: the number of non-dominated points (or rank 0 solutions) is typically very low at the beginning of the search, further on increases, and from a certain stage of the process the number of non-dominated solutions is nearly constant (i.e., equal to the population size). A constant value of p_{HCS} would hence lead to a permanent growth of the fraction of the local search within the memetic strategy, at least in the beginning of the search. To counteract to this effect, it seems to be better to start with a relatively high probability $p_{\max}$ and to decrease this value during the search process until a prescribed (low) probability $p_{\min}$ is reached. This value is then chosen for the remainder of the run of the algorithm.

TABLE VIII

NUMERICAL RESULTS OF NSGA-II AND THE MEMETIC STRATEGIES NSGA-II-HCS ON THE CONVEX PROBLEMS USING DIMENSION $n = 30$ AND PERFORMING 10 000 FUNCTION CALLS (MEAN VALUES AND STANDARD DEVIATIONS IN BRACKETS). STATISTICS WERE GATHERED FROM 30 INDEPENDENT RUNS

		Indicators	
Problems		GD	IGD
CONV1-U	NSGA-II(A)	1.2847(0.2258)	1.6994(0.2843)
	NSGA-II-HCS1(B)	0.5661(0.0938)	1.1123(0.4191)
	NSGA-II-HCS2(C)	0.0606(0.0054)	1.5931(0.9827)
	NSGA-II-HC2(D)	0.0590(0.0048)	1.3167(0.8445)
CONV1-C	NSGA-II(A)	1.3747(0.1687)	1.0594(0.1027)
	NSGA-II-HCS1(B)	0.1143(0.0417)	0.3470(0.0661)
	NSGA-II-HC2(C)	0.0063(0.0041)	0.0386(0.0540)
CONV2-U	NSGA-II(A)	2.1814(0.4247)	0.4618(0.0652)
	NSGA-II-HCS1(B)	1.1465(0.1249)	0.4533(0.0784)
	NSGA-II-HCS2(C)	0.1041(0.0133)	0.3815(0.1582)
	NSGA-II-HC2(D)	0.1171(0.0136)	0.3374(0.1466)
CONV2-C	NSGA-II(A)	2.1165(0.3591)	1.4540(0.1725)
	NSGA-II-HCS1(B)	0.4333(0.1673)	0.5873(0.0794)
	NSGA-II-HC2(C)	0.0127(0.0114)	0.4232(0.1127)

Set Coverage CONV1-U					
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$	$D \prec A$	$A \prec D$
0.9516(0.0977)	0.0218(0.0597)	0.9412(0.0825)	0.0032(0.0155)	0.9418(0.1144)	0(0)

Set Coverage CONV1-C			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
1(0)	0(0)	1(0)	0(0)

Set Coverage CONV2-U					
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$	$D \prec A$	$A \prec D$
0.8433(0.1572)	0.0103(0.0332)	0.9397(0.1233)	0(0)	0.9633(0.1035)	0(0)

Set Coverage CONV2-C			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
1(0)	0(0)	1(0)	0(0)

For the computations presented in the next section, we have used the following strategy which is based on the above considerations: starting with the probability $p_{HCS}(0) := p_{\max}$ the local search probabilities for the subsequent generations are updated as follows:

$$p_{HCS}(i) = \max\left\{ \frac{-2(p_{\max} - p_{\min})}{B} \sum_{j=1}^{i} fc(j) + p_{\max},\ p_{\min} \right\} \tag{26}$$

where $fc(j)$ denotes the number of function calls spent in the jth generation. Here, the first expression in (26) is a linear term in the number of function calls spent. Its value is $p_{\max}$ for zero function calls (i.e., at the beginning of the search) and $p_{\min}$ for $B, 2$. That is, after at least 50% of a given budget has been spent, the local search probability for future generations is constantly set to $p_{\min}$ (i.e., $p_{\min}$ times the population size is the number of HCS calls one is willing to spend in average per generation in advanced stages of the search).

2) SPEA2-HCS: Unlike above, where NSGA-II is used as base MOEA, we have observed that for a hybridization with SPEA2 it is not always beneficial to apply the HCS only to members of the archive which consists only of non-dominated solutions. This is because the archive can—in particular in early stages of the search—consist of few, and probably not well-spread solutions (which changes with increasing number of iterations). Thus, for a hybrid of HCS with SPEA2, we suggest to apply the local search operator to members of the mating pool, i.e., also to dominated solutions. Consequently, we propose by the above discussion to set p_{HCS} constant since the size of the mating pool does not change. See Algorithm 8 for a pseudocode of SPEA2-HCS.

V. RESULTS AND DISCUSSIONS

Here we present and discuss some numerical results for the HCS as well as for the two memetic strategies in order to demonstrate the strength of both the HCS as standalone algorithm as well as its benefit as a local search procedure within a given MOEA. The MOPs we have used here are listed in Table V. All computations have been done using the programming language MATLAB.[3]

A. HCS as a Standalone Algorithm

Since the two variants of the HCS as described in Algorithm 1 (which we will denote by HCS1 in this section) and in Algorithm 5 (denoted by HCS2) have no orientation in

[3]http://www.mathworks.com

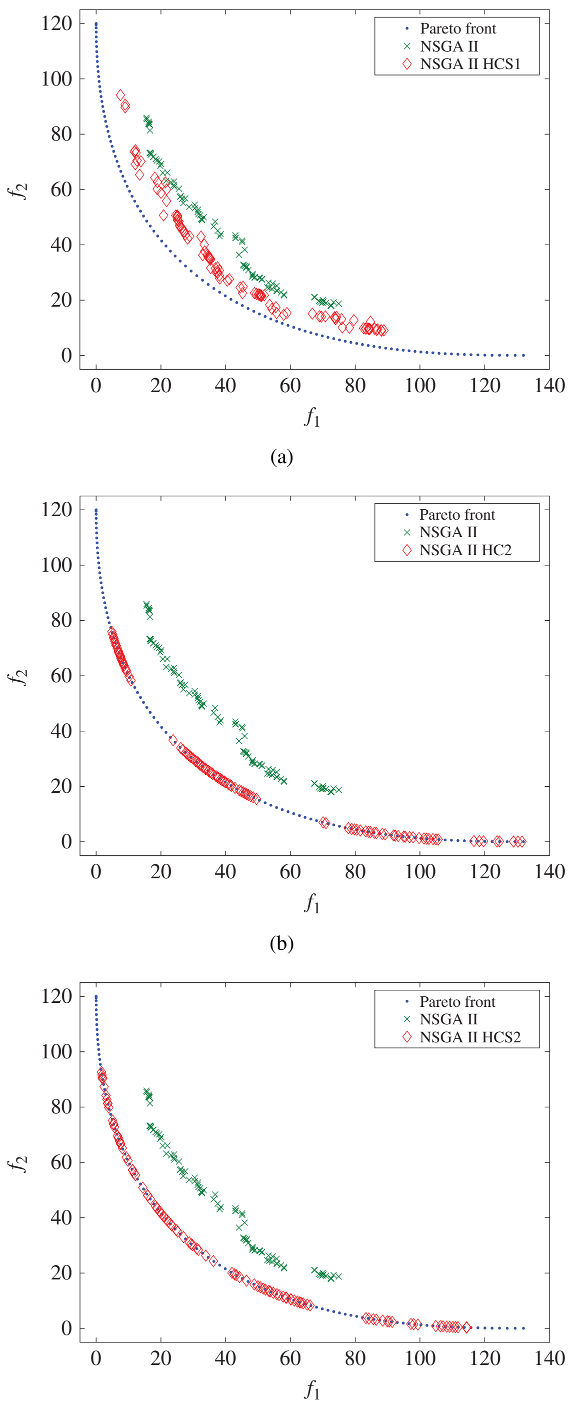


Fig. 6. Numerical result for CONV1-U using NSGA-II and the three memetic strategies NSGA-II-HCS1, NSGA-II-HC2, and NSGA-II-HCS2. The best result (spread *and* convergence) in this case was obtained by SPEA2-HCS2. (a) NSGA-II versus NSGA-II-HCS1, (b) NSGA-II versus NSGA-II-HC2, and (c) NSGA-II versus NSGA-II-HCS2.

the search along the Pareto set, we have modified it for bi-objective models in the following way in order to demonstrate its potential (see also discussion in Section III-A): the HCS—i.e., both variants—is started as described above. If the current iterate x_p is close enough to $\mathcal{P}$ such that the sidestep procedure can start (taking $N_s = 5$), first improvements according to f_1 are sought (leading to a "left up" movement from $F(x_{\text{temp}})$ along the Pareto front). If no improvements according to f_1 can be obtained, an analogous "right down" movement is performed starting again from x_{temp}. This is intended to "screen" the entire connected component of $\mathcal{P}$ which is near to x_{temp}.

However, since this orientation is not needed within the use of a MOEA because in that case only few iterates are being computed from a given starting point, and these modifications are only done within this section.

In the following we will test HCS1 and HCS2 on a convex model (i.e., a model that does not contain local minima where the local search can get stuck) and we will investigate both the unconstrained and the constrained case. Then we will consider a multimodal and constrained model (ZDT4).

Consider the MOP CONV1. The Pareto set of this model which is equal to $\mathcal{P}$ is located within $[-1, 1]^n$. First, we turn our attention to the unconstrained case: Fig. 3 shows two results obtained by the modified algorithms HCS1 and HCS2 with dimension $n = 10$ and domain $Q = [-5, 5]^{10}$. In both cases, the same starting point x_0 has been chosen. Since $\mathcal{P}$ is located within Q, no constraint handling techniques had to be applied in order to generate the sequence. For HCS1, a total of 1693 function calls had to be spent in order to get this result. For HCS2, 207 function calls, 60 evaluations of the gradient, and 192 evaluations of the Hessian were required (which are both given analytically. A conversion due to Section IV-B would lead to 13 095 function calls). Fig. 3 shows some qualitative differences as anticipated from the design of the different algorithms: HCS2 converges faster (in this case three iterates—not counting the function calls—were needed to reach $\mathcal{P}$, while HCS1 needed six iterations) and the non-dominated front is much better compared to the results obtained by the gradient free version HCS1 (in fact, the solution of HCS2 is practically identical to the true Pareto front). However, both results are satisfying since both non-dominated fronts represent a good approximation of the Pareto front with reasonable effort.

Next we consider the constrained case. Fig. 4 shows a numerical result from the HCS1 where we have used dimension $n = 2$ and for the domain $Q = [0.5, 1.5] \times [1, 2]$. The Pareto set is given by $P_Q = [0.5, 1] \times \{1\}$ and thus included in the boundary of Q. The figures show that also in this case the HCS1 is capable of approaching the solution set, and moving along it further on. However, a total of 997 function calls had to be spent in this setting, that is, more in comparison to the unconstrained case (note that the dimension of the model is much lower in the latter case).

Finally, we consider the problem ZDT4, which is a highly nonlinear and multimodal model. Fig. 5 shows two results in image space for two different initial solutions $x_0, z_0 \in Q = [0, 1] \times [-5, 5]^9$ and for the two variants of the HCS. As anticipated, the results for both algorithms and starting points differ significantly since the HCS is a local strategy and ZDT4 contains many local Pareto fronts. However, both procedures also in this case are able to explore a part of the local Pareto front which is located "near" to the image of the initial solution.

TABLE IX
COST OF THE HCS VARIANTS WITHIN SPEA2-HCS IN ONE RUN ON CONV1-U (THE RUN WHICH PRODUCED THE RESULT DISPLAYED IN FIG. 6)

Method	Hill climber calls (no sidestep)	Sidestep calls (w or w/o sidestep)	Function calls (total)
HCS1	7	68	2229
HC2	69	0	1262
HCS2	51	20	6145

B. HCS Coupled with a MOEA

Here we make some comparisons of the two state-of-the-art MOEAs NSGA-II and SPEA2 with their respective hybrid variants NSGA-II-HCS and SPEA2-HCS in order to demonstrate the possible benefit of the HCS when applied within a MOEA. Since we are dealing in this section with MOPs where the Pareto set is located at the boundary of the domain, we have used for these models a modification of HCS2 which acts just as a hill climber. That is, the search along the Pareto set is not performed (the value $\epsilon_{\mathcal{P}}$ is set to 0). To be conform with our notations and to avoid confusions, we denote this algorithm by HC2.

In order to evaluate the performance of the algorithms we have used the generational distance [66], the inverted generational distance [9] and the two set coverage measure [70] as indicators. A brief description follows.

Denote by δ_i the minimum Euclidean distance from the image $F(x_i)$ of a given point $x_i, i = 1, \ldots, n$, to the true Pareto front $F(P_Q)$. The generational distance (GD) of a set (population) $P = \{x_1, \ldots, x_n\}$ is defined as

$$GD = \frac{1}{n}\sqrt{\sum_{i=1}^{n} \delta_i^2}. \tag{27}$$

The inverted generational distance (IGD) is analogous to GD but measured from $F(P_Q)$ to $F(P)$.

Given two finite subsets A and B of $\mathbb{R}^n$ the two set coverage measure is defined as

$$SC(A, B) = A \prec B = \frac{|\{b \in B \text{ such that } \exists a \in A \text{ with } a \prec b\}|}{|B|}. \tag{28}$$

If $A \prec B = 1$, it means that all the elements of B are dominated by at least one element of A. If on the other hand $A \prec B = 0$, it means that no element of B is dominated by any element of A. Since the two set coverage metric is not symmetric always, both values $SC(A, B)$ and $SC(B, A)$ have to be taken into account.

1) Convex Models: First we consider the convex and thus unimodal models CONV1 ($k = 2$ objectives) and CONV2 ($k = 3$) which are taken from [15], and set the dimension of the parameter space to $n = 30$. For both models we are interested in the unconstrained case (where the Pareto set does not intersect with the boundary of the domain) and in the constrained case. Since the Pareto set of both unconstrained problems (that is, for $Q = \mathbb{R}^{30}$) is located within $[-1, 1]^{30}$ we have chosen to take the domains $Q_u := [-5, 5]^{30}$ and $Q_c = [-1, 1] \times [1, 2]^{29}$ for the unconstrained and the constrained model, respectively. Denote the resulting models by CONV1-U and CONV1-C (analogous to CONV2).

TABLE X
PARAMETER VALUES USED FOR SPEA2 AND NSGA-II AND THE MEMETIC STRATEGIES SPEA2-HCS AND NSGA-II-HCS ON THE DTLZ FUNCTIONS

Parameters	SPEA2-HCS	NSGA-II-HCS
N_{pop}	200	200
N_a	100	—
η_c	—	15
η_m	—	20
p_c	0.9	0.9
p_m	0.01	1/30
p_{HCS1}	0.3	—
p_{HCS2}	0.3	—
s	10	10
ε_y	1	5
$\varepsilon_{\mathcal{P}}$	0.0001	0.0001
r	0.05	0.1
Maxiter	5	10
N_{nd}	5	3
tol	10^{-4}	10^{-4}

For the realization we have used the parameter values displayed in Table VI and a budget of $B = 10\,000$ function calls. B is chosen relatively low in order to obtain significant differences in the indicator values. Tables VII and VIII show averaged numerical results obtained on the convex models using SPEA2, NSGA-II and the corresponding memetic strategies. For the two cases for which we hybridize the base MOEA with the HCS, the following observations can be made: the values of the Set Coverage and the GD improve when the HCS1 is used for additional local search. The values are even much better when using HC2 or HCS2 for the local search (the improvement is roughly one order of magnitude). The latter is certainly due to the fact that we are dealing with convex models: the gradient based search—though much more costly than HCS1 for $n = 30$, see Table III—leads to great improvements of given initial points which do not get stuck at local solutions. Thus, the population is pulled to the "right" set at any stage of the optimization process. For IGD, the results are not that conclusive; however, improvements can be observed.

In Fig. 6, one result of NSGA-II and their memetic variants is plotted which reflects the above discussion: Fig. 6(a) shows the effect of the HCS1, i.e., better convergence and spread than the result of NSGA-II due to the two search directions of HCS1. Convergence is on the other side much better in Fig. 6(b) and (c) where gradient information is used. When comparing the latter two figures, the effects of the sidestep get visible: The spread in Fig. 6(c) is apparently better than in Fig. 6(b), where the solutions fall into clusters.

Table IX gives an impression on the overall cost of the HCS within the search procedure. The table shows the amount of calls of the hill climber and sidestep procedures of HCS within SPEA2-HCS used to obtain the results in Fig. 6. HCS1 used 22% of the total budget B of SPEA2-HCS1. The relatively large amount of sidestep calls is due to the

TABLE XI

NUMERICAL RESULTS OF SPEA2 AND THE MEMETIC STRATEGIES SPEA2-HCS ON THE DTLZ BENCHMARKS USING DIMENSIONS $n = 30$, $k = 3$, AND PERFORMING 100 000 FUNCTION CALLS (MEAN VALUES AND STANDARD DEVIATIONS IN BRACKETS). STATISTICS WERE GATHERED FROM 30 INDEPENDENT RUNS

	Indicators		
Problems		GD	IGD
DTLZ1	SPEA2(A)	22.7984(1.6658)	2.4303(0.4716)
	SPEA2-HCS1(B)	12.2165(3.9738)	1.5389(0.2132)
	SPEA2-HC2(C)	48.06789(7.5157)	1.3770(0.4404)
DTLZ2	SPEA2(A)	0.0529(0.0102)	0.0011(0.0001)
	SPEA2-HCS1(B)	0.0342(0.0404)	0.0007(0.0001)
	SPEA2-HC2(C)	0.0557(0.0164)	0.0012(0.0001)
DTLZ3	SPEA2(A)	218.3484(9.9639)	10.3885(2.2814)
	SPEA2-HCS1(B)	28.1974(3.8434)	3.2152(0.3890)
	SPEA2-HCS2(C)	198.8947(27.5283)	3.6925(0.7106)

Set Coverage DTLZ1			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.9789(0.0286)	0.0157(0.0227)	0.6925(0.2840)	0.2100(0.2710)

Set Coverage DTLZ2			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.7820(0.0779)	0.0040(0.0089)	0.2060(0.0888)	0.3660(0.1740)

Set Coverage DTLZ3			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
1(0)	0(0)	0.7905(0.0862)	0.0877(0.0586)

low value of $N_{nd} (= 3)$. Larger values of N_{nd} would yield in less sidestep calls and in turn more hill climber calls. The cheapest local search operator is HC2 with a portion of 13% of the function calls since this operator merely computes the gradients to perform the hill-climber. The sidestep operator is much more costly since the second derivatives are involved as Table IX shows for HCS2: this local search operator spent 61% of B. The better result in Fig. 6(c) compared to Fig. 6(b) in terms of spread was obtained by merely 20 sidestep calls which, however, used quite a lot of function calls for this.

Concluding, it can be said that on the convex models (two and three objectives, constrained and unconstrained) a combination of the two base MOEAs with the HCS variants improves on one hand the overall performance of the search. On the other hand, considerations of the cost of the operators show that its application should be handled with care since else the HCS can take the lion's share of the budget which results in a risk to decrease the overall performance (note that we have assumed B to be constant).

2) DTLZ for Three Objectives: Finally, we consider the MOPs DTLZ1, DTLZ2, and DTLZ3 (see [14] and Table V), where we have chosen $n = 30$ for the dimension of the parameter space, $k = 3$ objectives, and the domain $Q = [0, 1]^{30}$ for all models.

Here, we have used the parameter values shown in Table X and a budget of $B_1 = 100\,000$ function calls for the multimodal models DTLZ1 and DTLZ3 and a budget of $B_2 = 10\,000$ for the unimodal model DTLZ2 (this has again been done in order to prevent too small values of the indicators). Tables XI and XII show averaged numerical results obtained on the test functions by the MOEAs and MEMOEAs under consideration. The conclusions which can be drawn from the results are not as straightforward as for the convex case. While a similar trend as for the convex case can be observed for DTLZ2, this does not hold for the multimodal models. Apparently, the two MEMOEAs which use HC2 cannot compete with their base MOEAs (however, to be fair the construction of the models already suggests that gradient information is worthless. Thus, it is rather a question of the choice of the model than an indication of a general failure of the HC2). On the other hand, SPEA2-HCS1 outperforms its base MOEA SPEA2 significantly on these highly multimodal models. Such an improvement, however, cannot be observed from NSGA-II-HCS1 and NSGA-II. The latter MOEA is already performing very well on these MOPs, which makes it hard for a local search strategy such as the HCS—which causes extra cost—to improve the overall performance, in particular on such complex problems.

Based on the numerical results presented in this section, it can be said that both variants of the standalone HCS accomplish their task, i.e., they are capable of moving toward and along the Pareto set. That is, by using the HCS, entire connected components of the (local) Pareto set can be explored starting with one single solution. Furthermore, it has been shown that the hybridization of the HCS with a given MOEAs can improve the overall performance of the base MOEA. More precisely, satisfying results have yet been obtained for unimodal MOPs. The results on multimodal models, however, are so far not that conclusive in general but depend on the base MOEA. Due to the relatively high cost of the HCS and the natural handicap of local search methods for multimodal problems, the balance of local and genetic search (such as for DTLZ1 and DTLZ3) is a challenging task. To handle this efficiently, adaptive strategies are indispensable, and further research should be done in that direction.

TABLE XII

NUMERICAL RESULTS OF NSGA-II AND THE MEMETIC STRATEGIES NSGA-II-HCS ON THE DTLZ BENCHMARKS USING DIMENSIONS $n = 30$, $k = 3$, AND PERFORMING 100 000 FUNCTION CALLS (MEAN VALUES AND STANDARD DEVIATIONS IN BRACKETS). STATISTICS WERE GATHERED FROM 30 INDEPENDENT RUNS

	Indicators		
Problems		GD	IGD
DTLZ1	NSGA-II(A)	8.5024(1.2081)	1.5998(0.1530)
	NSGA-II-HCS1(B)	10.4444(1.1505)	1.6856(0.2202)
	NSGA-II-HC2(C)	18.1555(4.6705)	1.0096(0.1899)
DTLZ2	NSGA-II(A)	0.0363(0.0041)	0.0022(0.0002)
	NSGA-II-HCS1(B)	0.0293(0.0033)	0.0018(0.0001)
	NSGA-II-HC2(C)	0.0097(0.0039)	0.0005(0)
DTLZ3	NSGA-II(A)	17.6126(3.5630)	3.2696(0.7721)
	NSGA-II-HCS1(B)	16.9381(2.9723)	2.9951(0.6861)
	NSGA-II-HC2(C)	24.6485(3.6724)	2.5127(0.4320)

Set Coverage DTLZ1			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.1610(0.1376)	0.6035(0.2565)	0.4885(0.1354)	0.3960(0.1627)

Set Coverage DTLZ2			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.7125(0.1869)	0.1235(0.0790)	0.9715(0.0284)	0.0105(0.0116)

Set Coverage DTLZ3			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.5010(0.3365)	0.3965(0.3243)	0.4435(0.3225)	0.4550(0.2857)

VI. CONCLUSION AND FUTURE WORK

We have proposed a novel point-wise iterative search procedure, called the HCS, which is designed for the local search of a given MOP. The HCS is intended to be capable to moving both toward and along the set of (local) Pareto points, depending on the location of the current iterate. We have proposed two variants of the HCS, a gradient free version (HCS1) and a version that exploits gradient information (HCS2). Both algorithms 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 extend. Further, we have addressed the problem of integrating the HCS into a given MOEA in order to obtain a novel memetic strategy. As examples, we have proposed the two algorithms (or family of algorithms) SPEA2-HCS and NSGA-II-HCS which are derived from SPEA2 and NSGA-II, respectively. Finally, we have presented some numerical results indicating the efficiency of the HCS as a standalone algorithm and its benefit when beeing integrated into a MOEA. More precisely, the results of SPEA2-HCS and NSGA-II-HCS show that the combination as proposed here is advantageous in many cases. However, it has to be mentioned that for this the design parameters of the HCS and the balance between local and genetic search have to be chosen properly, which is so far not done in an adaptive manner.

For future work, there are some interesting topics that can be addressed to advance the present work. In the first place, there is probably the design of an adaptive strategy to choose the design parameters as discussed above. Further, the coupling of local and global search could be improved. For this, an adaptive choice of the local search probability would be desirable (as done in [8]), or other techniques to reduce the computational complexity, e.g., by involving the information of the current population into the HCS. Another interesting topic would be to develop a version of HCS2 that can move efficiently along the Pareto set even if it is contained in the boundary of the domain which would allow for a more general use of the algorithm. Finally, it is conceivable to apply and advance the standalone HCS to other contexts. Areas which seem to be promising for that are numerical path-following (see [54] for a first study) or dynamic MOO.

REFERENCES

[1] S. F. Adra, I. Griffin, and P. J. Fleming, "Hybrid multiobjective genetic algorithm with a new adaptive local search process," in *Proc. Genetic Evol. Comput. Conf. (GECCO '05)*, vol. 1. New York: ACM, Jun. 2005, pp. 1009–1010.

[2] E. L. Allgower and K. Georg, *Numerical Continuation Methods*. 1st ed. New York: Springer-Verlag, 1990.

[3] L. Armijo, "Minimization of functions having Lipschitz-continuous first partial derivatives," *Pacific J. Math.*, vol. 16, no. 1, pp. 1–3, 1966.

[4] N. K. Bambha, S. S. Bhattacharyya, J. Teich, and E. Zitzler, "Systematic integration of parameterized local search into evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 8, no. 2, pp. 137–155, Apr. 2004.

[5] P. A. N. Bosman and E. D. de Jong, "Exploiting gradient information in numerical multiobjective evolutionary optimization," in *Proc. Genetic Evol. Comput. Conf. (GECCO '05)*, vol. 1. New York: ACM, Jun. 2005, pp. 755–762.

[6] J. Branke and S. Mostaghim, "About selecting the personal best in multiobjective particle swarm optimization," in *Proc. 9th Int. Conf. Parallel Problem Solving from Nature (PPSN)*, LNCS vol. 4193. Reykjavik, Iceland: Springer-Verlag, Sep. 2006, pp. 523–532.

[7] M. Brown and R. E. Smith, "Directed multiobjective optimisation," *Int. J. Comput., Syst. Signals*, vol. 6, no. 1, pp. 3–17, 2005.

[8] A. Caponio and F. Neri, "Integrating cross-dominance adaption in multiobjective memetic algorithms," *Multiobjective Memetic Algorithms*, SCI vol. 171. New York: Springer-Verlag, 2009, pp. 325–351.

[9] C. A. Coello Coello and N. Cruz Cortés, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming Evolvable Mach.*, vol. 6, no. 2, pp. 163–190, Jun. 2005.

[10] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multiobjective Problems*. 2nd ed. New York: Springer-Verlag, Sep. 2007.

[11] I. Das and J. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *SIAM J. Optimization*, vol. 8, no. 3, pp. 631–657, 1998.

[12] K. Deb, *Multiobjective Optimization Using Evolutionary Algorithms*. 1st ed. Chichester, U.K.: Wiley, 2001.

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA–II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[14] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, 1st ed. New York: Springer-Verlag, 2005, ch. 6, pp. 105–145.

[15] M. Dellnitz, O. Schütze, and T. Hestermeyer, "Covering Pareto sets by multilevel subdivision techniques," *J. Optimization Theory Applicat.*, vol. 124, no. 1, pp. 113–155, 2005.

[16] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. 1st ed. Englewood Cliffs, NJ: Prentice-Hall, 1983.

[17] J. Fliege and B. Fux Svaiter, "Steepest descent methods for multicriteria optimization," *Math. Methods Operations Res.*, vol. 51, no. 3, pp. 479–494, 2000.

[18] F. Glover and M. Laguna, *Tabu Search*. 1st ed. New York: Springer-Verlag, 1997.

[19] T. Goel and K. Deb, "Hybrid methods for multiobjective evolutionary algorithms," in *Proc. 4th Asia-Pacific Conf. Simulated Evol. Learning (SEAL '02)*, vol. 1. Singapore, Nov. 2002, pp. 188–192, Nanyang Tech. Univ.

[20] A. Griewank, "Evaluating derivatives: Principles and techniques of algorithmic diffierentiation," *Number 19 Frontiers Appl. Math.*, 1st ed. Philadelphia, PA: SIAM, 2000.

[21] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evol. Comput.*, vol. 9, no. 2, pp. 159–195, Summer 2001.

[22] K. Harada, J. Sakuma, and S. Kobayashi, "Uniform sampling of local pareto-optimal solution curves by pareto path following and its applications in multiobjective GA," in *Proc. Genetic Evol. Comput. Conf. (GECCO '07)*, vol. 1. London, U.K.: ACM, Jul. 2007, pp. 813–820.

[23] W. E. Hart, "Adaptive global optimization with local search," Ph.D. thesis, Dept. Comput. Sci. Eng., Univ. California, San Diego, CA, 1994.

[24] C. Hillermeier, *Nonlinear Multiobjective Optimization a Generalized Homotopy Approach*. 1st ed. Cambridge, MA: Birkhäuser, 2001.

[25] X. Hu, Z. Huang, and Z. Wang, "Hybridization of the multiobjective evolutionary algorithms and the gradient based algorithms," in *Proc. Congr. Evol. Comput. 2003 (CEC '03)*, vol. 2. Canberra, Australia: IEEE Press, Dec. 2003, pp. 870–877.

[26] C. Igel, N. Hansen, and S. Roth, "Covariance matrix adaptation for multiobjective optimization," *Evol. Comput.*, vol. 15, no. 1, pp. 1–28, Spring 2007.

[27] A. W. Iorio and X. Li, "Solving rotated multiobjective optimization problems using diffierential evolution," in *Proc. Advances Artificial Intell. (AI '04)*, LNAI vol. 3339. New York: Springer-Verlag, pp. 861–872.

[28] A. W. Iorio and X. Li, "Rotated problems and rotationally invariant crossover in evolutionary multiobjective optimization," *Int. J. Comput. Intell. Applicat.*, vol. 7, no. 2, pp. 149–186, 2008.

[29] H. Ishibuchi and T. Murata, "Multiobjective genetic local search algorithm," in *Proc. Int. Conf. Evol. Comput. 1996*, Nagoya, Japan: IEEE Press, pp. 119–124.

[30] H. Ishibuchi and T. Murata, "A multiobjective genetic local search algorithm and its application to flowshop scheduling," *IEEE Trans. Syst., Man, Cybern. Part C: Applicat. Rev.*, vol. 28, no. 3, pp. 392–403, Aug. 1998.

[31] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 204–223, Apr. 2003.

[32] A. Jaszkiewicz, "Do multiple-objective metaheuristics deliver on their promises? A computational experiment on the set-covering problem," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 133–143, Apr. 2003.

[33] W. E. Karush, "Minima of functions of several variables with inequalities as side conditions," Ph.D. thesis, Dept. Math., Univ. Chicago, Chicago, IL, 1939.

[34] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *New Ser. Sci.*, vol. 220, no. 4598, pp. 671–680, 1983.

[35] J. Knowles and D. Corne, "M-PAES: A memetic algorithm for multiobjective optimization," in *Proc. Congr. Evol. Comput. 2000*, vol. 1. Piscataway, NJ: IEEE Service Center, Jul. 2000, pp. 325–332.

[36] J. Knowles and D. Corne, "Memetic algorithms for multiobjective optimization: Issues, methods and prospects," *Recent Advances in Memetic Algorithms*, SFSC vol. 166. New York: Springer-Verlag, 2005, pp. 313–352.

[37] N. Krasnogor, "Studies on the theory and design space of memetic algorithms," Ph.D. thesis, Faculty Comput., Eng. Math. Sci., Univ. West England, Bristol, U.K., 2002.

[38] H. Kuhn and A. Tucker, "Nonlinear programming," in *Proc. 2nd Berkeley Symp. Math. Statist. Probability*, 1951, pp. 481–492.

[39] M. Lozano, F. Herrera, N. Krasnogor, and D. Molina, "Real-coded memetic algorithms with crossover hill-climbing," *Evol. Comput.*, vol. 12, no. 3, pp. 273–302, 2004.

[40] K. Miettinen, *Nonlinear Multiobjective Optimization*. 1st ed. Boston, MA: Kluwer, 1999.

[41] T. Murata, S. Kaige, and H. Ishibuchi, "Generalization of dominance relation-based replacement rules for memetic EMO algorithms," in *Proc. Part I Genetic Evol. Comput. (GECCO '03)*, LNCS vol. 2723. New York: Springer-Verlag, Jul. 2003 pp. 1234–1245.

[42] J. Nocedal and S. J. Wright, "Numerical optimization," *Springer Series in Operations Research*, New York: Springer-Verlag, 1999.

[43] U.-M. O'Reilly and F. Oppacher, "Hybridized crossover-based search techniques for program discovery," in *Proc. World Conf. Evol. Comput. 1995*, vol. 2. Perth, Australia: IEEE Press, pp. 573–578.

[44] V. Pareto, *Manual of Political Economy*. 1st ed. New York: MacMillan, 1971.

[45] S. Poles, E. Rigoni, and T. Robič, "MOGA-II performance on noisy optimization problems," in *Proc. Int. Conf. Bioinspired Optimization Methods Applicat. (BIOMA '04)*, Ljubljana, Slovenia, Oct. 2004, pp. 51–62.

[46] E. Rigoni and S. Poles, "NBI and MOGA-II, two complementary algorithms for multiobjective optimization," *Practical Approaches to Multiobjective Optimization*, Dagstuhl, Germany: Internationales Begegnungs-und Forschungszentrum für Informatik (IBFI), 2005, pp. 1–22.

[47] H. H. Rosenbrock, "An automatic method for finding the greatest or least value of a function," *Comput. J.*, vol. 3, no. 3, pp. 175–184, 1960.

[48] H. Satoh, M. Yamamura, and S. Kobayashi, "Minimal generation gap model for gas considering both exploration and exploitation," in *Proc. IIZUKA '96*, vol. 2. Fukuoka, Japan, 1996, pp. 494–497.

[49] S. Schäffler, R. Schultz, and K. Weinzierl, "A stochastic method for the solution of unconstrained vector optimization problems," *J. Optimization Theory Applicat.*, vol. 114, no. 1, pp. 209–222, 2002.

[50] O. Schütze, M. Laumanns, E. Tantar, C. A. Coello Coello, and E.-G. Talbi, "Convergence of stochastic search algorithms to gap-free Pareto front approximations," in *Proc. Genetic Evol. Comput. Conf. (GECCO '07)*, vol. 1. London, U.K.: ACM, Jul. 2007, pp. 892–899.

[51] O. Schütze, G. Sanchez, and C. A. Coello Coello, "A new memetic strategy for the numerical treatment of multiobjective optimization problems," in *Proc. Genetic Evol. Comput. Conf. (GECCO '08)*, Atlanta, GA: ACM, Jul. 2008, pp. 705–712.

[52] O. Schütze, "Set oriented methods for global optimization," Ph.D. thesis, Fakultät für Elektrotechnik, Informatik und Mathematik, Univ. Paderborn, Paderborn, Germany, 2004. Available: http://ubdata.uni-paderborn.de/ediss/17/2004/schuetze/

[53] O. Schütze, C. A. Coello Coello, S. Mostaghim, E.-G. Talbi, and M. Dellnitz, "Hybridizing evolutionary strategies with continuation methods for solving multiobjective problems," *Eng. Optimization*, vol. 40, no. 5, pp. 383–402, May 2008.

[54] O. Schütze, A. Lara, and C. A. Coello Coello, "Evolutionary continuation methods for optimization problems," in *Proc. Genetic Evol. Comput. Conf. (GECCO '09)*, Jul. 8–12, 2009, pp. 651–658.

[55] O. Schütze, S. Mostaghim, M. Dellnitz, and J. Teich, "Covering pareto sets by multilevel evolutionary subdivision techniques," in *Proc. 2nd Int. Conf. Evol. Multicriterion Optimization (EMO '03)*, LNCS. vol. 2632. Faro, Portugal: Springer-Verlag, Apr. 2003, pp. 118–132.

[56] O. Schütze, E.-G. Talbi, C. A. Coello Coello, L. V. Santana-Quintero, and G. Toscano Pulido, "A memetic PSO algorithm for scalar optimization problems," in *Proc. IEEE Swarm Intell. Symp. (SIS '07)*, Honolulu, HI: IEEE Press, Apr. 2007, pp. 128–134.

[57] K. Sindhya, K. Deb, and K. Miettinen, "A local search based evolutionary multiobjective optimization approach for fast and accurate convergence," in *Proc. Parallel Problem Solving from Nature–PPSN X*, LNCS vol. 5199. Dortmund, Germany: Springer-Verlag, Sep. 2008, pp. 815–824.

[58] A. Sinha, Y.-P. Chen, and D. E. Goldberg, "Designing efficient genetic and evolutionary algorithm hybrids," *Recent Advances Memetic Algorithms*, SFSC vol. 166. New York: Springer-Verlag, 2005, pp. 259–288.

[59] J. E. Smith and T. C. Fogarty, "Operator and parameter adaption in genetic algorithms," *Soft Comput.*, vol. 1, no. 2, pp. 81–87, 1997.

[60] O. Soliman, L. T. Bui, and H. Abbass, "A memetic coevolutionary multi-objective differential evolution algorithm," *Multiobjective Memetic Algorithms*, SCI vol. 171. New York: Springer-Verlag, 2009, pp. 325–351.

[61] B. Suman, "Study of simulated annealing based algorithms for multiobjective optimization of a constrained problem," *Comput. Chemical Eng.*, vol. 28, no. 9, pp. 1849–1871, Aug. 2004.

[62] M. Vasile, "A behavior-based meta-heuristic for robust global trajectory optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC '07)*, Singapore: IEEE Press, pp. 494–497.

[63] M. Vasile, "Hybrid behavioral-based multiobjective space trajectory optimization," *Multiobjective Memetic Algorithms*, SCI vol. 171. New York: Springer-Verlag, 2009, pp. 231–254.

[64] M. Vasile and M. Locatelli, "A hybrid multiagent approach for global trajectory optimization," *J. Global Optimization*, vol. 44, no. 4, pp. 461–479, Aug. 2009.

[65] M. Vasile and C. Maddock, "Design of optimal spacecraft-asteroid formations through a hybrid global optimization approach," *Int. J. Intelligent Comput. Cybern.*, vol. 1, no. 2, pp. 239–268, 2008.

[66] D. A. Van Veldhuizen, "Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations," Ph.D. thesis, Dept Elect. Comput. Eng., Graduate School Eng., Air Force Inst. Technol., Wright-Patterson AFB, Dayton, OH, May 1999.

[67] Y. Wang, Z. Cai, G. Guo, and Y. Zhou, "Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems," *IEEE Trans. Syst., Man Cybern. Part B–Cybern.*, vol. 37, no. 3, pp. 560–575, Jun. 2007.

[68] E. F. Wanner, F. G. Guimaraes, R. H. C. Takahashi, and P. J. Fleming, "A quadratic approximation-based local search procedure for multiobjective genetic algorithms," in *Proc. IEEE Congr. Evol. Comput. (CEC '06)*, Vancouver, BC, Canada: IEEE, Jul. 2006, pp. 3361–3368.

[69] Y. Yin, T. Okabe, and B. Sendhoff, "Adapting weighted aggregation for multiobjective evolution strategies," in *Proc. 1st Int. Conf. Proc. Evol. Multicriterion Optimization (EMO '01)*, LNCS vol. 1993. New York: Springer-Verlag, pp. 96–110.

[70] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, Jun. 2000.

[71] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm," in *Proc. Evol. Methods Design, Optimization Control Applicat. Ind. Problems (EUROGEN '01)*, Athens, Greece, 2002, pp. 95–100.

[72] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

Adriana Lara received the B.Sc. degree in physics and mathematics in 2001 from the National Polytechnic Institute of México. In 2003, she received the M.Sc. degree in electrical and computer engineering, from the Departamento de Computación, Centro de Investigación y de Estudios Avanzados-Instituto Politécnico Nacional (CINVESTAV-IPN), México City, Mexico. She is currently pursuing the Ph.D. degree in computer sciences at CINVESTAV-IPN.

Earlier, she was an Associate Professor at the National Polytechnic Institute of México, México City. Her main research interests include multiobjective optimization, gradient based techniques, and hybrid algorithms.

Gustavo A. Sánchez received the B.S. degree in 1994 from the University of Metz, France, and the M.S. degree in 2004 from Simon Bolivar University, Caracas, Venezuela, both in systems engineering.

He was a Control Engineer for several oil companies. He is currently an Associate Professor in the Department of Process and Systems, Simon Bolivar University, Caracas. His research interests include control systems, multiobjective optimization and evolutionary algorithms.

Carlos A. Coello Coello (M'99–SM'04) received the B.Sc. degree in civil engineering from the Universidad Autónoma de Chiapas, México, and the M.Sc. and Ph.D. degrees in computer science from Tulane University, New Orleans, LA, in 1991, 1993, and 1996, respectively.

He is currently a Professor (CINVESTAV-3-D Researcher) at the Department of Computación, Centro de Investigación y de Estudios Avanzados-Instituto Politécnico Nacional (CINVESTAV-IPN), México City, Mexico. He has authored and co-authored over 180 technical papers and several book chapters. He has also co-authored the book *Evolutionary Algorithms for Solving Multiobjective Problems* (2nd edition, Springer, 2007). His major research interests are evolutionary multiobjective optimization and constraint-handling techniques for evolutionary algorithms.

Dr. Coello received the 2007 National Research Award from the Mexican Academy of Sciences in the area of exact sciences. He is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and serves on the Editorial Board of seven other international journals. He also chairs the IEEE Computationally Intelligent Society Task Force on Multiobjective Evolutionary Algorithms. He is a member of the ACM, Sigma Xi, and the Mexican Academy of Sciences.

Oliver Schütze received the Diploma in mathematics from the University of Bayreuth, Germany, and the Ph.D. degree in natural sciences from the University of Paderborn, Germany, in 1999 and 2004, respectively.

He is currently a Professor (CINVESTAV-2C Researcher) at the Departamento de Computación, Centro de Investigación y de Estudios Avanzados-Instituto Politécnico Nacional (CINVESTAV-IPN), México City, Mexico.