HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization

Johannes Bader johannes.bader@tik.ee.ethz.ch
Computer Engineering and Networks Laboratory, ETH Zurich, 8092 Zurich, Switzerland

Eckart Zitzler eckart.zitzler@phbern.ch
PHBern, University of Teacher Education, Institute for Continuing Professional Education, CH-3006 Bern, Switzerland

Abstract

In the field of evolutionary multi-criterion optimization, the hypervolume indicator is the only single set quality measure that is known to be strictly monotonic with regard to Pareto dominance: whenever a Pareto set approximation entirely dominates another one, then the indicator value of the dominant set will also be better. This property is of high interest and relevance for problems involving a large number of objective functions. However, the high computational effort required for hypervolume calculation has so far prevented the full exploitation of this indicator's potential; current hypervolume-based search algorithms are limited to problems with only a few objectives.

This paper addresses this issue and proposes a fast search algorithm that uses Monte Carlo simulation to approximate the exact hypervolume values. The main idea is not that the actual indicator values are important, but rather that the rankings of solutions induced by the hypervolume indicator. In detail, we present HypE, a hypervolume estimation algorithm for multi-objective optimization, by which the accuracy of the estimates and the available computing resources can be traded off; thereby, not only do many-objective problems become feasible with hypervolume-based search, but also the runtime can be flexibly adapted. Moreover, we show how the same principle can be used to statistically compare the outcomes of different multi-objective optimizers with respect to the hypervolume—so far, statistical testing has been restricted to scenarios with few objectives. The experimental results indicate that HypE is highly effective for many-objective problems in comparison to existing multi-objective evolutionary algorithms.

HypE is available for download at http://www.tik.ee.ethz.ch/sop/download/supplementary/hype/.

Keywords

Hypervolume indicator, multi-objective optimization, multi-objective evolutionary algorithm, Monte Carlo sampling.

1 Motivation

The vast majority of studies in the field of evolutionary multi-objective optimization (EMO) are concerned with the following set problem: find a set of solutions that as a whole represents a good approximation of the Pareto-optimal set. To this end, the original multi-objective problem consisting of

- the decision space X,
- the objective space $Z = \mathbb{R}^n$,
- a vector function $f = (f_1, f_2, \ldots, f_n)$ comprising n objective functions $f_i : X \to \mathbb{R}$, which are without loss of generality to be minimized, and
- a relation $\leq$ on Z, which induces a preference relation $\preceq$ on X with $a \preceq b :\Leftrightarrow f(a) \leq f(b)$ for $a, b \in X$,

is usually transformed into a single-objective set problem (Zitzler et al., 2008). The search space Ψ of the resulting set problem includes all possible Pareto set approximations,[1] that is, Ψ contains all multisets over X. The preference relation $\preceq$ can be used to define a corresponding set preference relation $\preccurlyeq$ on Ψ where

$$A \preccurlyeq B :\Leftrightarrow \forall b \in B \, \exists a \in A : \; a \preceq b \tag{1}$$

for all Pareto set approximations $A, B \in \Psi$. In the following, we will assume that weak Pareto dominance is the underlying preference relation (cf. Zitzler et al., 2008).[2]

A key question when tackling such a set problem is how to define the optimization criterion. Many multi-objective evolutionary algorithms (MOEAs) implement a combination of Pareto dominance on sets and a diversity measure based on Euclidean distance in the objective space, for example, NSGA-II (Deb et al., 2000) and SPEA2 (Zitzler et al., 2002). While these methods have been successfully employed in various bi-objective optimization scenarios, they appear to have difficulties when the number of objectives increases (Wagner et al., 2007). As a consequence, researchers have tried to develop alternative concepts, and a recent trend is to use set quality measures, also denoted as quality indicators, for search—so far, they have mainly been used for performance assessment. Of particular interest in this context is the hypervolume indicator (Zitzler and Thiele, 1998a, 1999) as it is the only quality indicator known to be fully sensitive to Pareto dominance—a property especially desirable when multi-objective functions are involved.

Several hypervolume-based MOEAs have been proposed (e.g., Emmerich et al., 2005; Igel et al., 2007; Brockhoff and Zitzler, 2007), but their main drawback is their extreme computational overhead. Although there have been recent studies presenting improved algorithms for hypervolume calculation, currently high-dimensional problems with six or more objectives are infeasible for these MOEAs. Therefore, the question is whether and how fast hypervolume-based search algorithms can be designed that exploit the advantages of the hypervolume indicator and at the same time are scalable with respect to the number of objectives.

[1]Here, a Pareto set approximation may also contain dominated solutions as well as duplicates, in contrast to the notation in Zitzler et al. (2003).

[2]For reasons of simplicity, we will use the term "u weakly dominates v" resp. "u dominates v" independently of whether u and v are elements of X, Z, or Ψ. For instance, A weakly dominates b with $A \in \Psi$ and $b \in X$ means $A \preccurlyeq \{b\}$ and a dominates z with $a \in X$ and $z \in Z$ means $f(a) \leq z \wedge z \not\leq f(a)$.

A first attempt in this direction has been presented in Bader et al. (2010). The main idea is to estimate—by means of Monte Carlo simulation—the ranking of the individuals that is induced by the hypervolume indicator and not to determine the exact indicator values. This paper proposes an advanced method called HypE (hypervolume estimation algorithm for multi-objective optimization) that is based on the same idea, but uses more effective fitness assignment and sampling strategies. In detail, the main contributions of this work can be summarized as follows:

1. A novel method to assign fitness values to individuals based on the hypervolume indicator—for both mating and environmental selection;
2. A hypervolume-based search algorithm (HypE) using Monte Carlo simulation that can be applied to problems with arbitrarily many objectives;
3. A statistical testing procedure that allows us to compare the outcomes of different multi-objective optimizers with respect to the hypervolume indicator in many-objective scenarios.

As we will show in the following, the proposed search algorithm can be easily tuned regarding the available computing resources and the number of objectives involved. Thereby, it opens a new perspective on how to treat many-objective problems, and the presented concepts may also be helpful for other types of quality indicators to be integrated in the optimization process.

2 A Brief Review of Hypervolume-Related Research

The hypervolume indicator was originally proposed and employed by Zitzler and Thiele (1998b, 1999) to quantitatively compare the outcomes of different MOEAs. In these two first publications, the indicator was denoted as "size of the space covered," and later also other terms such as "hyperarea metric" (Van Veldhuizen, 1999), "S-metric" (Zitzler, 1999), "hypervolume indicator" (Zitzler et al., 2003), and hypervolume measure (Beume, Naujoks, et al., 2007) were used. Besides the names, there are also different definitions available, based on polytopes (Zitzler and Thiele, 1999), the Lebesgue measure (Laumanns et al., 1999; Knowles, 2002; Fleischer, 2003), or the attainment function (Zitzler et al., 2007).

As to hypervolume calculation, the first algorithms (Zitzler, 2001; Knowles, 2002) operated recursively and in each recursion step the number of objectives was decremented; the underlying principle is known as the hypervolume by slicing objectives approach (While et al., 2006). While the method used by Zitzler and Thiele (1998b, 1999) was never published (only the source code is publicly available, Zitzler, 2001), Knowles (2002) independently proposed and described a similar method. A few years later, this approach was systematically studied for the first time and heuristics to accelerate the computation were proposed in the study of While et al. (2005, 2006). All these algorithms have a worst-case runtime complexity that is exponential in the number of objectives, more specifically $\mathcal{O}(N^{n-1})$ where N is the number of solutions considered (Knowles, 2002; While et al., 2006). A different approach was presented by Fleischer (2003) who mistakenly claimed a polynomial worst-case runtime complexity; and While (2005) showed that it is exponential in n as well. Recently, advanced algorithms for hypervolume calculation have been proposed, a dimension-sweep method (Fonseca et al., 2006) with a worst-case runtime complexity of $\mathcal{O}(N^{n-2} \log N)$, and a specialized algorithm related to the Klee measure problem (Beume and Rudolph, 2006) the runtime of which is in the worst case of order $\mathcal{O}(N \log N + N^{n/2})$. Furthermore, Yang and Ding (2007) described an algorithm for which they claim a worst-case runtime complexity of $\mathcal{O}((n/2)^N)$. The fact that there is no exact polynomial algorithm available gave rise to the hypothesis that this problem in general is hard to solve, although the tightest known lower bound is of order $\Omega(N \log N)$ (Beume et al., 2007). New results substantiate this hypothesis: Bringmann and Friedrich (2008) have proven that the problem of computing the hypervolume is $\#P$-complete, that is, it is expected that no polynomial algorithm exists since this would imply $NP = P$.

The complexity of the hypervolume calculation in terms of programming and computation time may explain why this measure was seldom used until 2003. However, this changed with the advent of theoretical studies that provided evidence for a unique property of this indicator (Knowles and Corne, 2002; Zitzler et al., 2003; Fleischer, 2003): it is the only indicator known to be strictly monotonic with respect to Pareto dominance and thereby guaranteeing that the Pareto-optimal front achieves the maximum hypervolume possible, while any worse set will be assigned a worse indicator value. This property is especially desirable with many-objective problems and since classical MOEAs have been shown to have difficulties in such scenarios (Wagner et al., 2007), a trend can be observed in the literature to directly use the hypervolume indicator for search.

Knowles and Corne were the first to propose the integration of the hypervolume indicator into the optimization process (Knowles, 2002; Knowles and Corne, 2003). In particular, they described a strategy to maintain a separate, bounded archive of nondominated solutions based on the hypervolume indicator. Huband et al. (2003) presented an MOEA which includes a modified SPEA2 environmental selection procedure where a hypervolume-related measure replaces the original density estimation technique. In the work of Zitzler and Künzli (2004), the binary hypervolume indicator was used to compare individuals and to assign corresponding fitness values within a general indicator-based evolutionary algorithm (IBEA). The first MOEA tailored specifically to the hypervolume indicator was described by Emmerich et al. (2005); it combines nondominated sorting with the hypervolume indicator and considers one offspring per generation (steady state). Similar fitness assignment strategies were later adopted by Zitzler et al. (2007) and Igel et al. (2007), and also other search algorithms were proposed where the hypervolume indicator is partially used for search guidance (Nicolini, 2005; Mostaghim et al., 2007). Moreover, specific aspects such as hypervolume-based environmental selection (Bradstreet et al., 2006, 2007; Bradstreet, 2009; see also Section 3.2), and explicit gradient determination for hypervolume landscapes (Emmerich et al., 2007) have been investigated recently.

To date, the hypervolume indicator is one of the most popular set quality measures. For instance, almost one fourth of the papers published in the proceedings of the EMO 2007 conference (Obayashi et al., 2007) report on the use of or are dedicated to the hypervolume indicator. However, there are still two major drawbacks that current research acitivities try to tackle: (i) the high computation effort and (ii) the bias of the indicator in terms of user preferences. The former issue has been addressed in different ways: by automatically reducing the number of objectives (Brockhoff and Zitzler, 2007) and by approximating the indicator values using Monte Carlo methods (Everson et al., 2002; Bader et al., 2010). Everson et al. (2002) used a basic Monte Carlo technique for performance assessment in order to estimate the values of the binary hypervolume indicator (Zitzler, 1999); with their approach the error ratio is not polynomially bounded. In contrast, the scheme presented in Bringmann and Friedrich (2008) is a fully polynomial randomized approximation scheme where the error ratio is polynomial in the input size. The issue of statistically comparing hypervolume estimates was not addressed in these two papers. Another study by Bader et al. (2010)—a precursor study for the present paper—employed Monte Carlo simulation for fast hypervolume-based search. As to the bias issue, first proof-of-principle results were presented by Zitzler et al. (2007) that demonstrate this, and also show how the hypervolume indicator can be adapted to different user preferences.

3 Hypervolume-Based Fitness Assignment

When considering the hypervolume indicator as the objective function of the underlying set problem, the main question is how to make use of this measure within a multi-objective optimizer to guide the search. In the context of an MOEA, this refers to selection and one can distinguish two situations:

1. The selection of solutions to be varied (mating selection).
2. The selection of solutions to be kept in memory (environmental selection).

Since the indicator as such operates on (multi)sets of solutions, while selection considers single solutions, a strategy for assigning fitness values to solutions is required. Most hypervolume-based algorithms first perform a nondominated sorting and then rank solutions within a particular front according to the hypervolume loss that results from the removal of a specific solution (Knowles and Corne, 2003; Emmerich et al., 2005; Igel et al., 2007; Bader et al., 2010). In the following, we propose a generalized fitness assignment strategy that takes into account the entire objective space weakly dominated by a population. We will first provide a basic scheme for mating selection and then present an extension for environmental selection. Afterward, we briefly discuss how the fitness values can be computed exactly using a slightly modified hypervolume calculation algorithm.

3.1 Basic Scheme for Mating Selection

To begin with, we formally define the hypervolume indicator as a basis for the following discussions. Different definitions can be found in the literature, and here we use the one from Zitzler et al. (2008) which draws upon the Lebesgue measure as proposed by Laumanns et al. (1999) and considers a reference set of objective vectors.

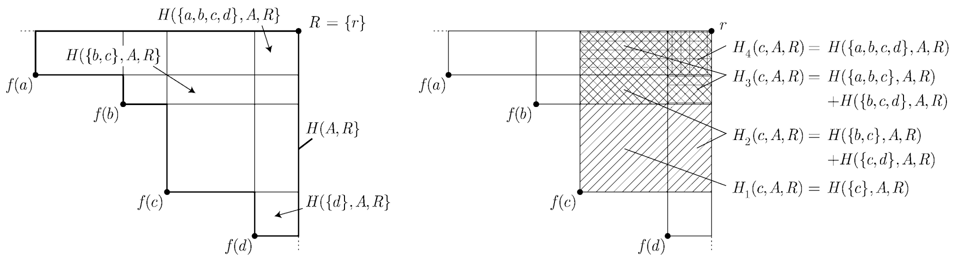
DEFINITION 1: *Let $A \in \Psi$ be a Pareto set approximation and $R \subset Z$ be a reference set of mutually nondominating objective vectors. Then the hypervolume indicator I_H can be defined as*

$$I_H(A, R) := \lambda(H(A, R)) \tag{2}$$

where

$$H(A, R) := \{z \in Z\,;\, \exists a \in A\, \exists r \in R : f(a) \leq z \leq r\} \tag{3}$$

and λ is the Lebesgue measure with $\lambda(H(A, R)) = \int_{\mathbb{R}^n} \mathbf{1}_{H(A,R)}(z)dz$ and $\mathbf{1}_{H(A,R)}$ being the characteristic function of $H(A, R)$.


$H(\{a,b,c,d\},A,R)$
$R=\{r\}$
$H(\{b,c\},A,R)$
$f(a)$
$f(b)$
$H(A,R)$
$f(c)$
$H(\{d\},A,R)$
$f(d)$
r
$H_4(c,A,R)=H(\{a,b,c,d\},A,R)$
$H_3(c,A,R)=H(\{a,b,c\},A,R)+H(\{b,c,d\},A,R)$
$H_2(c,A,R)=H(\{b,c\},A,R)+H(\{c,d\},A,R)$
$H_1(c,A,R)=H(\{c\},A,R)$
$f(a)$
$f(b)$
$f(c)$
$f(d)$

(a) The relationship between $H(A, R)$ and $H(S, A, R)$

(b) The relationship between $H(S, A, R)$ and $H_i(a, A, R)$

Figure 1: Illustration of the notions of $H(A, R)$, $H(S, A, R)$, and $H_i(a, A, R)$ in the objective space for a Pareto set approximation $A = \{a, b, c, d\}$ and reference set $R = \{r\}$.

The set $H(A, R)$ denotes the set of objective vectors that are enclosed by the front $f(A)$ given by A and the reference set R.

The subspace $H(A, R)$ of Z can be further split into partitions $H(S, A, R)$, each associated with a specific subset $S \subseteq A$:

$$H(S, A, R) := \Big[\bigcap_{s\in S} H(\{s\}, R)\Big] \Big\backslash \Big[\bigcup_{a\in A\setminus S} H(\{a\}, R)\Big]. \tag{4}$$

The set $H(S, A, R) \subseteq Z$ represents the portion of the objective space that is jointly weakly dominated by the solutions in S and not weakly dominated by any other solution in A. It holds

$$\dot{\bigcup_{S\subseteq A}} H(S, A, R) = H(A, R) \tag{5}$$

which is illustrated in Figure 1(a). That the partitions are disjoint can be easily shown: Assume that there are two nonidentical subsets S_1, S_2 of A for which $H(S_1, A, R) \cap H(S_2, A, R) \neq \emptyset$; since the sets are not identical, there exists with loss of generality an element $a \in S_1$ which is not contained in S_2; from the above definition it follows that $H(\{a\}, R) \supseteq H(S_1, A, R)$ and therefore $H(\{a\}, R) \cap H(S_2, A, R) \neq \emptyset$; the latter statement leads to a contradiction since $H(\{a\}, R)$ cannot be part of $H(S_2, A, R)$ when $a \notin S_2$.

In practice, it is infeasible to determine all distinct $H(S, A, R)$ due to combinatorial explosion. Instead, we consider a more compact splitting of the dominated objective space that refers to single solutions:

$$H_i(a, A, R) := \bigcup_{\substack{S\subseteq A\\ a\in S\\ |S|=i}} H(S, A, R). \tag{6}$$

According to this definition, $H_i(a, A, R)$ stands for the portion of the objective space that is jointly and solely weakly dominated by a and any $i - 1$ further solutions from A (see Figure 1(b)). Note that the sets $H_1(a, A, R), H_2(a, A, R), \ldots, H_{|A|}(a, A, R)$ are disjoint for

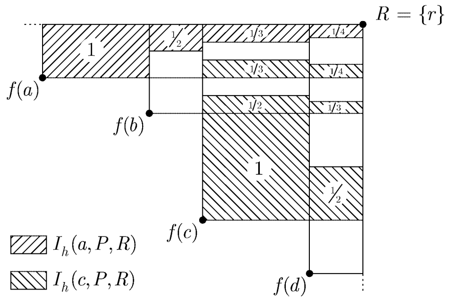


Figure 2: Illustration of the basic fitness assignment scheme where the fitness F_a of a solution a is set to $F_a = I_h(a, P, R)$.

a given $a \in A$, that is, $\dot{\bigcup}_{1 \leq i \leq |A|} H_i(a, A, R) = H(\{a\}, R)$, while the sets $H_i(a, A, R)$ and $H_i(b, A, R)$ may be overlapping for fixed i and different solutions $a, b \in A$. This slightly different notion has reduced the number of subspaces to be considered from $2^{|A|}$ for $H(S, A, R)$ to $|A|^2$ for $H_i(a, A, R)$.

Now, given an arbitrary population $P \in \Psi$ one obtains for each solution a contained in P a vector $(\lambda(H_1(a, P, R)), \lambda(H_2(a, P, R)), \ldots, \lambda(H_{|P|}(a, P, R)))$ of hypervolume contributions. These vectors can be used to assign fitness values to solutions; Section 3.3 describes how the corresponding values $\lambda(H_i(a, A, R))$ can be computed. While most hypervolume-based search algorithms only take the first components, that is, $\lambda(H_1(a, P, R))$, into account, we propose the following scheme to aggregate the hypervolume contributions into a single scalar value.

Definition 2: *Let $A \in \Psi$ and $R \subset Z$. Then the function I_h with*

$$I_h(a, A, R) := \sum_{i=1}^{|A|} \frac{1}{i} \lambda(H_i(a, A, R)) \tag{7}$$

gives for each solution $a \in A$ the hypervolume that can be attributed to a with regard to the overall hypervolume $I_H(A, R)$.

The motivation behind this definition is simple: the hypervolume contribution of each partition $H(S, A, R)$ is shared equally among the dominating solutions $s \in S$. That means the portion of Z solely weakly dominated by a specific solution a is fully attributed to a, the portion of Z that a weakly dominates together with another solution b is attributed half to a and so forth (this principle is illustrated in Figure 2). Thereby, the overall hypervolume is distributed among the distinct solutions according to their hypervolume contributions as the following theorem shows (the proof can be found in the technical report by Bader and Zitzler, 2008). Note that this scheme does not require that the solutions of the considered Pareto set approximation A are mutually nondominating; it applies to nondominated and dominated solutions alike.

Theorem 1: *Let $A \in \Psi$ and $R \subset Z$. Then it holds*

$$I_H(A, R) = \sum_{a \in A} I_h(a, A, R) \tag{8}$$

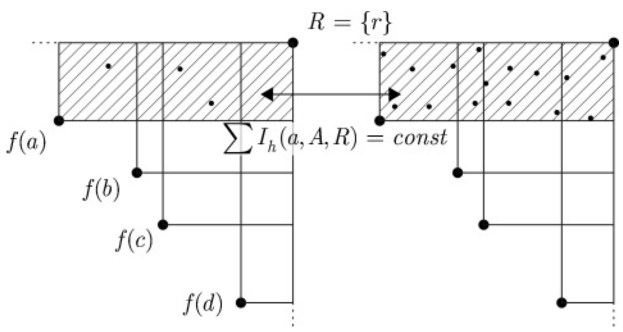


Figure 3: Shows for an example population the selection probabilities for the population members (left). The sizes of the points correlate with the corresponding selection probabilities. As one can see on the right, the overall selection probability for the shaded area does not change when dominated solutions are added to the population.

This aggregation method has some desirable properties that make it well suited to mating selection where the fitness F_a of a population member $a \in P$ is $F_a = I_h(a, P, R)$ and the corresponding selection probability p_a equals $F_a/I_H(P, R)$. As Figure 3 demonstrates, the accumulated selection probability remains the same for any subspace $H(\{a\}, R)$ with $a \in P$, independently of how many individuals $b \in P$ are mapped to $H(\{a\}, R)$ and how the individuals are located within $H(\{a\}, R)$. This can be formally stated in the next theorem; the proof can again be found in Bader and Zitzler (2008).

THEOREM 2: *Let $A \in \Psi$ and $R \subset Z$. For every $a \in A$ and all multisets $B_1, B_2 \in \Psi$ with $\{a\} \preccurlyeq B_1$ and $\{a\} \preccurlyeq B_2$ holds*

$$\sum_{b_1 \in \{a\} \cup B_1} I_h(b_1, \{a\} \cup B_1, R) = \sum_{b_2 \in \{a\} \cup B_2} I_h(b_2, \{a\} \cup B_2, R). \tag{9}$$

Since the selection probability per subspace is constant as long as the overall hypervolume value does not change, adding dominated solutions to the population leads to a redistribution of the selection probabilities and thereby implements a natural niching mechanism. Another advantage of this fitness assignment scheme is that it takes all hypervolume contributions $H_i(a, P, R)$ for $1 \leq i \leq |P|$ into account. As will be discussed in Section 4, this allows us to more accurately estimate the ranking of the individuals according to their fitness values when using Monte Carlo simulation.

In order to study the usefulness of this fitness assignment strategy, we consider the following experiment. A standard evolutionary algorithm implementing pure nondominated sorting fitness is applied to a selected test function (bi-objective WFG1 (Huband et al., 2006) using the setting as described in Section 6) and run for 100 generations. Then mating selection is carried out on the resulting population, that is, the individuals are reevaluated using the fitness scheme under consideration and offspring are generated employing binary tournament selection with replacement and corresponding variation operators. The hypervolume of the (multi)set of offspring is taken as an indicator for the effectiveness of the fitness assignment scheme. By comparing the resulting hypervolume values for different strategies [constant fitness leading to uniform selection, nondominated sorting plus $\lambda(H_1(a, P, R))$, and the proposed fitness according to

Table 1: Comparison of three fitness assignment schemes: (1) constant fitness, (2) nondominated sorting plus $\lambda(H_1(a, P, R))$, and (3) the proposed method. Each value gives the percentage of cases where the method associated with that row yields a higher hypervolume value than the method associated with the corresponding column.

	Constant (1)	Standard (2)	New (3)
Constant (1)	—	44%	28%
Standard (2)	56%	—	37%
New (3)	72%	63%	—

Definition 2] and for 100 repetitions of this experiment, we can investigate the influence of the fitness assignment strategy on the mating selection process.

The Quade test, a modification of Friedman's test which has more power when comparing few treatments (Conover, 1999), reveals that there are significant differences in the quality of the generated offspring populations at a significance level of 0.01 (test statistics: $T_3 = 12.2$). Performing post hoc pairwise comparisons following Conover (1999) using the same significance level as in the Quade test provides evidence that the proposed fitness strategy can have advantages over the other two strategies, as can be seen in Table 1; in the considered setting, the hypervolume values achieved are significantly better. The standard hypervolume-based fitness significantly outperforms the constant fitness. Note that the required computation resources need to be taken into account. In practice, this means that the advantage over uniform selection may diminish when fitness computation becomes expensive. This aspect will be investigated in Section 6.

Next, we will extend and generalize the fitness assignment scheme with regard to the environmental selection phase.

3.2 Extended Scheme for Environmental Selection

In the context of hypervolume-based multi-objective search, environmental selection can be formulated in terms of the hypervolume subset selection problem (HSSP).

Definition 3: *Let $A \in \Psi$, $R \subset Z$, and $k \in \{0, 1, \ldots, |A|\}$. The hypervolume subset selection problem (HSSP) is defined as the problem of finding a subset $A' \subseteq A$ with $|A'| = |A| - k$ such that the overall hypervolume loss is minimized, that is,*

$$I_H(A', R) = \max_{\substack{A'' \subseteq A \\ |A''| = |A| - k}} I_H(A'', R). \tag{10}$$

Here, we assume that parents and offspring have been merged into a single population P which then needs to be truncated by removing k solutions. Since dominated solutions in the population do not affect the overall hypervolume, they can be deleted first; therefore, we assume in the following that all solutions in P are incomparable[3] or indifferent[4] to each other.

[3]Two solutions $a, b \in X$ are called *incomparable* if and only if neither weakly dominates the other one, that is, $a \not\preceq b$ and $b \not\preceq a$.

[4]Two solutions $a, b \in X$ are called *indifferent* if and only if both weakly dominate each other, that is, $a \preceq b$ and $b \preceq a$.

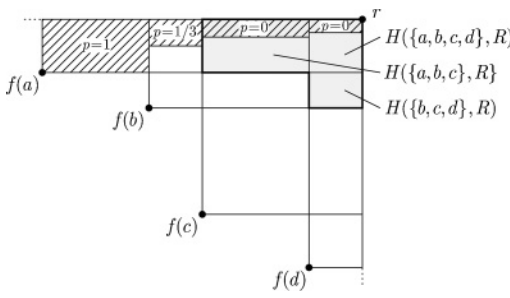


Figure 4: The figure is based on the previous example with $A = \{a, b, c, d\}$, $R = \{r\}$. The figure shows (i) which portion of the objective space remains dominated if any two solutions are removed from A (shaded area), and (ii) the probability p that a particular area that can be attributed to $a \in A$ is lost if a is removed from A together with any other solution in A.

If $k = 1$, then HSSP can be solved exactly by removing that solution a from the population P with the lowest value $\lambda(H_1(a, P, R))$; this is the principle implemented in most hypervolume-based MOEAs which consider one offspring per generation (e.g., Knowles and Corne, 2003; Emmerich et al., 2005; Igel et al., 2007). However, it has been recently shown that exchanging only one solution in the population as is done with steady state MOEAs ($k = 1$) may lead to premature convergence to a local optimum in the hypervolume landscape (Zitzler et al., 2008). This problem can be avoided when generating at least as many offspring as parents are available, that is, $k \geq |P|/2$.

For arbitrary values of k, dynamic programming can be used to solve HSSP in a bi-objective setting; in the presence of three or more objectives, it is an open problem whether HSSP becomes NP-hard. In practice, a greedy heuristic is employed to obtain an approximation (Zitzler and Künzli, 2004; Brockhoff and Zitzler, 2007): all solutions are evaluated with respect to their usefulness and the l least important solutions are removed where l is a prespecified parameter. Most popular are the following two approaches:

1. **Iterative ($l = 1$):** The greedy heuristic is applied k times in a row; each time, the worst solution is removed and afterward the remaining solutions are re-evaluated.
2. **One shot ($l = k$):** The greedy heuristic is only applied once; the solutions are evaluated and the k worst solutions are removed in one step.

Best results are usually obtained using the iterative approach, as the reevaluation increases the quality of the generated approximation. In contrast, the one-shot approach substantially reduces the computation effort, but the quality of the resulting subset is lower. In the context of density-based MOEAs, the first approach is for instance used in SPEA2, while the second is employed in NSGA-II.

The key issue with respect to the above greedy strategy is how to evaluate the usefulness of a solution. The scheme presented in Definition 2 has the drawback that portions of the objective space are taken into account that definitely will not change. Consider, for instance, a population with four solutions as shown in Figure 4; when two solutions need to be removed ($k = 2$), then the subspaces $H(\{a, b, c\}, P, R)$, $H(\{b, c, d\}, P, R)$, and $H(\{a, b, c, d\}, P, R)$ remain weakly dominated independently of which solutions are deleted. This observation led to the idea of considering the expected loss in hypervolume that can be attributed to a particular solution when exactly k solutions are removed. In detail, we consider for each $a \in P$ the average hypervolume loss over all subsets $S \subseteq P$ that contain a and $k-1$ further solutions; this value can be easily computed by slightly extending the scheme from Definition 2 as follows.

DEFINITION 4: *Let $A \in \Psi$, $R \subset Z$, and $k \in \{0, 1, \dots, |A|\}$. Then the function I_h^k with*

$$I_h^k(a, A, R) := \frac{1}{|\mathcal{S}|} \sum_{S \in \mathcal{S}} \left[\sum_{\substack{T \subseteq S \\ a \in T}} \frac{1}{|T|} \lambda(H(T, A, R)) \right] \tag{11}$$

where $\mathcal{S} = \{S \subseteq A\,;\, a \in S \,\wedge\, |S| = k\}$ contains all subsets of A that include a and have cardinality k gives for each solution $a \in A$, the expected hypervolume loss that can be attributed to a when a and $k-1$ uniformly randomly chosen solutions from A are removed from A.

Note that $I_h^1(a, A, R) = \lambda(H_1(a, A, R))$ and $I_h^{|A|}(a, A, R) = I_h(a, A, R)$, that is, this modified scheme can be regarded as a generalization of the scheme presented in Definition 2 and the commonly used fitness assignment strategy for hypervolume-based search (Knowles and Corne, 2003; Emmerich et al., 2005; Igel et al., 2007; Bader et al., 2010). The next theorem shows how to calculate $I_h^k(a, A, R)$ without averaging over all subsets $S \in \mathcal{S}$; the proof can be found in Bader and Zitzler (2008).

THEOREM 3: *Let $A \in \Psi$, $R \subset Z$, and $k \in \{0, 1, \dots, |A|\}$. Then it holds*

$$I_h^k(a, A, R) = \sum_{i=1}^{k} \frac{\alpha_i}{i} \lambda(H_i(a, A, R)) \quad \textit{where} \quad \alpha_i := \prod_{j=1}^{i-1} \frac{k-j}{|A|-j}. \tag{12}$$

Next, we will study the effectiveness of $I_h^k(a, A, R)$ for approximating the optimal HSSP solution. To this end, we assume that for the iterative greedy strategy ($l = 1$) in the first round the values $I_h^k(a, A, R)$ are considered, in the second round the values $I_h^{k\text{-}1}(a, A, R)$, and so forth; each time an individual assigned the lowest value is selected for removal. For the one-step greedy method ($l = k$), only the $I_h^k(a, A, R)$ values are considered.

Table 2 provides a comparison of the different techniques for 100,000 randomly chosen Pareto set approximations $A \in \Psi$ containing 10 incomparable solutions, where the 10 points are randomly distributed on a three dimensional unit simplex, that is, we consider a three objective scenario. The parameter k was set to 5, so that half of the solutions needed to be removed. The relatively small numbers were chosen so that the optimal subsets could be computed by enumeration. Thereby, the maximum hypervolume values achievable could be determined.

The comparison reveals that the new fitness assignment scheme is in the considered scenario more effective in approximating HSSP than the standard scheme. The mean relative distance (see Table 2) to the optimal solution is about 60% smaller than the distance achieved using I_h^1 in the iterative case and about 44% smaller in the one shot case. Furthermore, the optimum was found much more often in comparison to the standard fitness: 34% more often for the iterative approach and 497% in the one shot scenario.

Finally, note that the proposed evaluation function I_h^k will be combined with non-dominated sorting for environmental selection (cf. Section 5), similar to work done

Table 2: Comparison of greedy strategies for the HSSP (iterative vs. one shot) using the new (I_h^k) and the standard hypervolume fitness (I_h^1); as a reference, purely random deletions are considered as well. The first column gives the portion of cases in which an optimal subset was generated; the second column provides the average difference in hypervolume between optimal and the generated subset. The last two columns reflect the direct comparisons between the two fitness schemes for each greedy approach (iterative, one shot) separately; they give the percentages of cases where the corresponding method was better than or equal to the other one.

Greedy strategy	Optimum found (%)	Distance	Better (%)	Equal (%)
Iterative with I_h^k	59.8	1.09×10^{-3}	30.3	66.5
Iterative with I_h^1	44.5	2.59×10^{-3}	3.17	66.5
One shot with I_h^k	16.9	39.3×10^{-3}	65.2	23.7
One shot with I_h^1	3.4	69.6×10^{-3}	11.1	23.7
Uniformly random	0.381	257×10^{-3}		

elsewhere (Emmerich et al., 2005; Igel et al., 2007; Brockhoff and Zitzler, 2007; Zitzler et al., 2007; Bader et al., 2010). One reason is computation time: with nondominated sorting, the worst dominated solutions can be removed quickly without invoking the hypervolume calculation algorithm; this advantage mainly applies to low-dimensional problems and to the early stage of the search process. Another reason is that the full benefits of the scheme proposed in Definition 4 can be exploited when the Pareto set approximation A under consideration only contains incomparable and indifferent solutions; otherwise, it cannot be guaranteed that nondominated solutions are preferred over dominated ones.

3.3 Exact Calculation of I_h^k

In this subsection, we tackle the question of how to calculate the fitness values for a given population $P \in \Psi$. We present an algorithm that determines the values $I_h^k(a, P, R)$ for all elements $a \in P$ and a fixed k. In the case of mating selection, k equals $|P|$, and in the case of environmental selection, k is the number of solutions to be removed from P. The algorithm operates according to the hypervolume by slicing objectives principle (Zitzler, 2001; Knowles, 2002; While et al., 2006), but differs from existing methods in that it allows: (i) for the consideration of a set R of reference points and (ii) for all fitness values, to be computed, for example, the $I_h^1(a, P, R)$ values for $k = 1$, in parallel for any number of objectives instead of subsequently as in the work of Beume, Naujoks, et al. (2007). Although it looks at all partitions $H(S, P, R)$ with $S \subseteq P$ explicitly, the worst case runtime complexity is not affected by this; it is of order $\mathcal{O}(|P|^n + n|P| \log |P|)$, assuming that sorting of the solutions in all dimensions is carried out as a preprocessing step. Please note that faster hypervolume calculation algorithms exists, most notably the algorithm by Beume and Rudolph (2006).[5] Clearly, the algorithm is only feasible for a low number of objectives, and the next section discusses how the fitness values can be estimated using Monte Carlo methods.

[5]Adjusting this method to the fitness measure I_h^k is not straightforward, hence only the extension of the basic HSO approach is demonstrated here. The main focus of the paper is to compare the algorithms with respect to fixed numbers of generations, while in all runtime results reported the hypervolume algorithms are comparable *relative* to one another, benefiting to the same extent from faster implementations.

Algorithm 1 Hypervolume-Based Fitness Value Computation

```
Require: population $P \in \Psi$, reference set $R \subseteq Z$, fitness parameter $k \in \mathbb{N}$
 1: procedure computeHypervolume($P, R, k$)
 2:     $\mathcal{F} \leftarrow \bigcup_{a \in P} \{(a, 0)\}$
 3:     return doSlicing($\mathcal{F}, R, k, n, 1, (\infty, \infty, \ldots, \infty)$);
 4: end procedure
```

Algorithm 2 Recursive Objective Space Partitioning

```
Require: current fitness assignment $\mathcal{F}$, reference set $R \subseteq Z$, fitness parameter $k \in \mathbb{N}$,
    recursion level $i$, partial volume $V \in \mathbb{R}$, scan positions $(z_1, \ldots, z_n) \in \mathbb{R}^n$
 1: procedure doSlicing($\mathcal{F}, R, k, i, V, (z_1, \ldots, z_n)$)
 2:                                    /*filter out relevant solutions and reference points*/
 3:     $UP \leftarrow \bigcup_{(a,v) \in \mathcal{F},\ \forall i < j \leq n:\ f_j(a) \leq z_j} \{f(a)\}$
 4:     $UR \leftarrow \bigcup_{(r_1,\ldots,r_n) \in R,\ \forall i < j \leq n:\ r_j \geq z_j} \{(r_1, \ldots, r_n)\}$
 5:     if $i = 0 \wedge UR \neq \emptyset$ then                    /*end of recursion reached*/
 6:         $\alpha \leftarrow \prod_{j=1}^{|UP|-1} (k - j)/(|\mathcal{F}| - j)$
 7:         $\mathcal{F}' \leftarrow \emptyset$
 8:         for all $(a, v) \in \mathcal{F}$ do          /*update hypervolumes of filtered solutions*/
 9:             if $\forall 1 \leq j \leq n : f_j(a) \leq z_j$ then
10:                 $\mathcal{F}' \leftarrow \mathcal{F}' \cup \{(a, v + \frac{\alpha}{|UP|} V)\}$
11:             else
12:                 $\mathcal{F}' \leftarrow \mathcal{F}' \cup \{(a, v)\}$
13:             end if
14:         end for
15:     else if $i > 0$ then                              /*recursion continues*/
16:         $\mathcal{F}' \leftarrow \mathcal{F}$
17:         $U \leftarrow UP \cup UR$
18:         while $U \neq \emptyset$ do            /*scan current dimension in ascending order*/
19:             $u^* \leftarrow \min_{(u_1,\ldots,u_n) \in U} u_i$
20:             $U' \leftarrow \{(u_1, \ldots, u_n) \in U \mid u_i > u^*\}$
21:             if $U' \neq \emptyset$ then
22:                 $V' = V \cdot \left((\min_{(u'_1,\ldots,u'_n) \in U'} u'_i) - u^*\right)$
23:                 $\mathcal{F}' \leftarrow$ doSlicing($\mathcal{F}', R, k, i - 1, V', (z_1, \ldots, z_{i-1}, u^*, z_{i+1}, \ldots, z_n)$ )
24:             end if
25:             $U = U'$
26:         end while
27:     end if
28:     return $\mathcal{F}'$
29: end procedure
```

Details of the procedure are given by Algorithms 1 and 2. Algorithm 1 just provides the top level call to the recursive function *doSlicing* and returns a fitness assignment $\mathcal{F}$, a multiset containing for each $a \in P$ a corresponding pair (a, v) where v is the fitness value. Note that n at Line 3 denotes the number of objectives. Algorithm 2 recursively cuts the dominated space into hyperrectangles and returns a (partial) fitness assignment

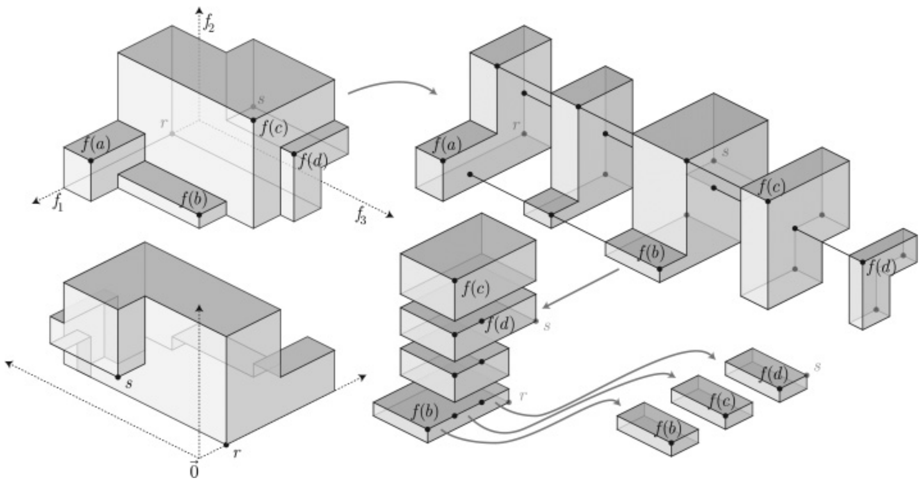


Figure 5: Illustration of the principle underlying Algorithm 2 where one looks from $(-\infty, -\infty, -\infty)$ on the front except for the lower left picture where one looks from $(\infty, -\infty, \infty)$ to the origin. First, the dominated polytope is cut along the third dimension leading to five slices, which are again cut along the second dimension and finally along the first dimension. In contrast to existing hypervolume by slicing objectives algorithms, dominated points are also carried along.

$\mathcal{F}'$. At each recursion level, a scan is performed along a specific objective—given by i—with u^* representing the current scan position. The vector $(z_1, \ldots, z_n)$ contains for all dimensions the scan positions, and at each invocation of *doSlicing* solutions (more precisely: their objective vectors) and reference points are filtered out according to these scan positions (Lines 3 and 4) where dominated solutions may also be selected, in contrast to previous algorithms (Zitzler, 2001; Knowles, 2002; While et al., 2006). Furthermore, the partial volume V is updated before recursively invoking Algorithm 2 based on the distance to the next scan position. At the lowest recursion level ($i = 0$), the variable V gives the hypervolume of the partition $H(A, P, R)$, that is, $V = \lambda(H(A, P, R))$ where A stands for the remaining solutions fulfilling the bounds given by the vector $(z_1, \ldots, z_n)$, and UP contains the objective vectors corresponding to A (see Line 3). Since the fitness according to Definition 4 is additive with respect to the partitions, for each $a \in A$ the partial fitness value v can be updated by adding $\frac{\alpha_{|UP|}}{|UP|}V$. Note that the population is a multiset, that is, it may contain indifferent solutions or even duplicates; therefore, all the other sets in the algorithms are multisets.

The following example illustrates the working principle of the hypervolume computation.

Example 1: Consider the three-objective scenario depicted in Figure 5 where the population contains four solutions a, b, c, d the objective vectors of which are $f(a) = (-10, -3, -2)$, $f(b) = (-8, -1, -8)$, $f(c) = (-6, -8, -10)$, and $f(d) = (-4, -5, -11)$, and the reference set includes two points $r = (-2, 0, 0)$, $s = (0, -3, -4)$. Furthermore, let the parameter k be 2.

In the first call of *doSlicing*, it holds $i = 3$ and U contains all objective vectors associated with the population and all reference points. The following representation shows U with its elements sorted in ascending order according to their third vector components:

$$U = \begin{array}{r@{\,:\,}l} f(d) & (-4, -5, -11) \downarrow \\ f(c) & (-6, -8, -10) \\ f(b) & (-8, -1, -8) \\ s & (-0, -3, -4) \\ f(a) & (-10, -3, -2) \\ r & (-2, 0, 0) \end{array} \tag{13}$$

Hence, in the first two iterations of the loop beginning at Line 18 the variable u^* is assigned to $f_3(d) = -11$ resp. $u^* = f_3(c) = -10$. Within the third iteration, U is reduced to $\{f(a), f(b), r, s\}$ which yields $u^* = f_3(b) = -8$ and in turn $V' = 1 \cdot (-4 - (-8)) = 4$ with the current vector of scan positions being $(z_1, z_2, z_3) = (\infty, \infty, -8)$; these values are passed to the next recursion level $i = 2$ where U is initialized at Line 17 as follows (this time sorted according to the second dimension):

$$U = \begin{array}{r@{\,:\,}l} f(c) & (-6, -8, -10) \downarrow \\ f(d) & (-4, -5, -11) \\ s & (0, -3, -4) \\ f(b) & (-8, -1, -8) \\ r & (-2, 0, 0) \end{array} \tag{14}$$

Now, after three iterations of the loop at Line 18 with $u^* = f_2(c) = -8$, $u^* = f_2(d) = -5$, and $u^* = s_2 = -3$, respectively, U is reduced in the fourth iteration to $\{f(b), r\}$ and u^* is set to $f_2(b) = -1$. As a result, $V' = 1 \cdot 4 \cdot (0 - (-1)) = 4$ and $(z_1, z_2, z_3) = (\infty, -1, -8)$ which are the parameters for the next recursive invocation of *doSlicing* where U is set to:

$$U = \begin{array}{r@{\,:\,}l} f(b) & (-8, -1, -8) \downarrow \\ f(c) & (-6, -8, -10) \\ f(d) & (-4, -5, -11) \\ r & (-2, 0, 0) \end{array} \tag{15}$$

At this recursion level with $i = 1$, in the second iteration it holds $u^* = f_1(c) = -6$ and $V' = 1 \cdot 4 \cdot 1 \cdot (-4 - (-6)) = 8$. When calling *doSlicing* at this stage, the last recursion level is reached ($i = 0$): First, α is computed based on the population size $N = 4$, the number of individuals dominating the hyperrectangle ($|UP| = 2$), and the fitness parameter $k = 2$, which yields $\alpha = 1/3$; then for b and c, the fitness values are increased by adding $\alpha \cdot V/|UP| = 4/3$.

Applying this procedure to all slices at a particular recursion level identifies all hyperrectangles which constitute the portion of the objective space enclosed by the population and the reference set.

4 Estimating Hypervolume Contributions Using Monte Carlo Simulation

As outlined above, the computation of the proposed hypervolume-based fitness scheme is so expensive that only problems with a maximum of four or five objectives are tractable within reasonable time limits. However, in the context of randomized search heuristics, one may argue that the exact fitness values are not crucial and approximated values may be sufficient; furthermore, if using pure rank-based selection schemes, then only the resulting order of the individuals matters. These considerations lead to the idea of estimating the hypervolume contributions by means of Monte Carlo simulation.

In order to approximate the fitness values according to Definition 4, we need to estimate the Lebesgue measures of the domains $H_i(a, P, R)$ where $P \in \Psi$ is the population. Since these domains are all integrable, their Lebesgue measure can be approximated by means of Monte Carlo simulation.

For this purpose, a sampling space $S \subseteq Z$ has to be defined with the following properties: (i) the hypervolume of S can easily be computed, (ii) samples from the space S can be generated quickly, and (iii) S is a superset of the domains $H_i(a, P, R)$ the hypervolumes of which one would like to approximate. The latter condition is met by setting $S = H(P, R)$, but since it is hard both to calculate the Lebesgue measure of this sampling space and to draw samples from it, we propose using the axis-aligned minimum bounding box containing the $H_i(a, P, R)$ subspaces instead, as in the following

$$S := \{(z_1, \ldots, z_n) \in Z \mid \forall 1 \leq i \leq n : l_i \leq z_i \leq u_i\} \tag{16}$$

where

$$l_i := \min_{a \in P} f_i(a) \qquad\qquad u_i := \max_{(r_1,\ldots,r_n) \in R} r_i \tag{17}$$

for $1 \leq i \leq n$. Hence, the volume V of the sampling space S is given by $V = \prod_{i=1}^{n} \max\{0, u_i - l_i\}$.

Now given S, sampling is carried out by selecting M objective vectors $s_1, \ldots, s_M$ from S uniformly at random. For each s_j the algorithm checks whether it lies in any partition $H_i(a, P, R)$ for $1 \leq i \leq k$ and $a \in P$. This can be determined in two steps: first, it is verified that s_j is "below" the reference set R, that is, there exists $r \in R$ that is dominated by s_j; second, it is verified that the multiset A of those population members dominating s_j is not empty. If both conditions are fulfilled, then we know that—given A—the sampling point s_j lies in all partitions $H_i(a, P, R)$ where $i = |A|$ and $a \in A$. This situation will be denoted as a *hit* regarding the ith partition of a. If any of the two above conditions is not fulfilled, then we call s_j a *miss*. Let $X_j^{(i,a)}$ denote the corresponding random variable that is equal to 1 in case of a hit of s_j regarding the ith partition of a and 0 otherwise.

Based on the M sampling points, we obtain an estimate for $\lambda(H_i(a, P, R))$ by simply counting the number of hits and multiplying the hit ratio with the volume of the sampling box

$$\hat{\lambda}\big(H_i(a, P, R)\big) = \frac{\sum_{j=1}^{M} X_j^{(i,a))}}{M} \cdot V. \tag{18}$$

This value approaches the exact value $\lambda(H_i(a, P, R))$ with increasing M by the law of large numbers. Due to the linearity of the expectation operator, the fitness scheme according to Equation (11) can be approximated by replacing the Lebesgue measure with the respective estimates given by Equation (18):

$$\hat{I}_h^k(a, P, R) = \sum_{i=1}^{k} \frac{\alpha_i}{i} \cdot \left(\frac{\sum_{j=1}^{M} X_j^{(i,a))}}{M} V \right). \tag{19}$$

Algorithm 3 Hypervolume-based Fitness Value Estimation

```
Require: population $P \in \Psi$, reference set $R \subseteq Z$, fitness parameter $k \in \mathbb{N}$, number of
    sampling points $M \in \mathbb{N}$
 1: procedure estimateHypervolume($P, R, k, M$)
 2:     for $i \leftarrow 1, n$ do                                /*determine sampling box S*/
 3:         $l_i = \min_{a \in P} f_i(a)$
 4:         $u_i = \max_{(r_1,\ldots,r_n) \in R} r_i$
 5:     end for
 6:     $S \leftarrow [l_1, u_1] \times \cdots \times [l_n, u_n]$
 7:     $V \leftarrow \prod_{i=1}^{n} \max\{0, (u_i - l_i)\}$
 8:     $\mathcal{F} \leftarrow \bigcup_{a \in P} \{(a, 0)\}$                          /*reset fitness assignment*/
 9:     for $j \leftarrow 1, M$ do                                     /*perform sampling*/
10:         choose $s \in S$ uniformly at random
11:         if $\exists r \in R : \; s \leq r$ then
12:             $UP \leftarrow \bigcup_{a \in P, \; f(a) \leq s} \{f(a)\}$
13:             if $|UP| \leq k$ then                        /*hit in a relevant partition*/
14:                 $\alpha \leftarrow \prod_{l=1}^{|UP|-1} \frac{k-l}{|P|-l}$
15:                 $\mathcal{F}' \leftarrow \emptyset$
16:                 for all $(a, v) \in \mathcal{F}$ do          /*update hypervolume estimates*/
17:                     if $f(a) \leq s$ then
18:                         $\mathcal{F}' \leftarrow \mathcal{F}' \cup \{(a, v + \frac{\alpha}{|UP|} \cdot \frac{V}{M})\}$
19:                     else
20:                         $\mathcal{F}' \leftarrow \mathcal{F}' \cup \{(a, v)\}$
21:                     end if
22:                 end for
23:                 $\mathcal{F} \leftarrow \mathcal{F}'$
24:             end if
25:         end if
26:     end for
27:     return $\mathcal{F}$
28: end procedure
```

The details of the estimation procedure are described by Algorithm 3 which returns a fitness assignment, that is, for each $a \in P$ the corresponding hypervolume estimate $\hat{I}_h^k(a, P, R)$. It will be used later by the EA presented in Section 5. Note that the partitions $H_i(a, P, R)$ with $i > k$ do not need to be considered for the fitness calculation as they do not contribute to the I_h^k values that we would like to estimate (refer to Definition 4).

In order to study how closely the sample size M and the accuracy of the estimates are related, a simple experiment was carried out: 10 imaginary individuals $a \in A$ were generated, the objective vectors $f(a)$ of which are uniformly distributed at random on a three dimensional unit simplex, similar to the experiments presented in Table 2. These individuals were then ranked on the one hand according to the estimates $\hat{I}_h^{|A|}$ and on the other hand with respect to the exact values $I_h^{|A|}$. The closer the former ranking is to the latter ranking, the higher is the accuracy of the estimation procedure given by Algorithm 3. In order to quantify the differences between the two rankings, we calculated the percentage of all pairs (i, j) with $1 \leq i < j \leq |A|$ where the individuals

Table 3: Accuracy of the ranking of 10 individuals according to $\hat{I}_h^{10}$ (19) in comparison to I_h^{10} for different sample sizes. The percentages represent the number of pairs of individuals ranked correctly.

Number of samples M	Ranking accuracy (%)
10^1	56.0
10^2	74.1
10^3	89.9
10^4	96.9
10^5	99.2
10^6	99.8
10^7	100.0

Algorithm 4 HypE Main Loop

```
Require: reference set $R \subseteq Z$, population size $N \in \mathbb{N}$, number of generations $g_{\max}$, number of sampling points $M \in \mathbb{N}$
1: initialize population $P$ by selecting $N$ solutions from $X$ uniformly at random
2: $g \leftarrow 0$
3: while $g \leq g_{\max}$ do
4:     $P' \leftarrow matingSelection(P, R, N, M)$
5:     $P'' \leftarrow variation(P', N)$
6:     $P \leftarrow environmentalSelection(P \cup P'', R, N, M)$
7:     $g \leftarrow g + 1$
8: end while
```

at the ith position and the jth position in the ranking according to $I_h^{|A|}$ have the same order in the ranking according to $\hat{I}_h^{|A|}$ (see Scharnow et al., 2004). The experiment was repeated for different numbers of sampling points as shown in Table 3. The experimental results indicate that 10,000 samples are necessary to achieve an error below 5% and that 10,000,000 sampling points are sufficient in this setting to obtain the exact ranking.

Seeing the close relationship between sample size and accuracy, one may ask whether M can be adjusted automatically on the basis of confidence intervals. In the technical report of Bader and Zitzler (2008), confidence intervals are derived for the sampled fitness values. Based on these, an adaptive version of the sampling procedure is presented and compared to the strategy using a fixed number of samples.

5 HypE: Hypervolume Estimation Algorithm for Multi-Objective Optimization

In this section, we describe an evolutionary algorithm named HypE, which is based on the fitness assignment schemes presented in the previous sections. When the number of objectives is small ($\leq$3), the hypervolume values I_h^k are computed exactly using Algorithm 1; otherwise they are estimated based on Algorithm 3.

The main loop of HypE is given by Algorithm 4. It reflects a standard evolutionary algorithm and consists of the successive application of mating selection (Algorithm 5), variation, and environmental selection (Algorithm 6). As to mating selection, binary tournament selection is proposed here, although any other selection scheme could be

Algorithm 5 HypEMating Selection

```
Require: population $P \in \Psi$, reference set $R \subseteq Z$, number of offspring $N \in \mathbb{N}$, number
    of sampling points $M \in \mathbb{N}$
 1: procedure matingSelection($P, R, N, M$)
 2:     if $n \leq 3$ then
 3:         $\mathcal{F} \leftarrow$ computeHypervolume($P, R, N$)
 4:     else
 5:         $\mathcal{F} \leftarrow$ estimateHypervolume($P, R, N, M$)
 6:     end if
 7:     $Q \leftarrow \emptyset$
 8:     while $|Q| < N$ do
 9:         choose $(a, v_a), (b, v_b) \in \mathcal{F}$ uniformly at random
10:         if $v_a > v_b$ then
11:             $Q \leftarrow Q \cup \{a\}$
12:         else
13:             $Q \leftarrow Q \cup \{b\}$
14:         end if
15:     end while
16:     return $Q$
17: end procedure
```

used as well. The procedure *variation* encapsulates the application of mutation and recombination operators to generate N offspring. Finally, environmental selection aims at selecting the most promising N solutions from the multiset-union of parent population and offspring; more precisely, it creates a new population by carrying out the following two steps:

1. First, the union of parents and offspring is divided into disjoint partitions using the principle of nondominated sorting (Goldberg, 1989; Deb et al., 2000), also known as dominance depth. Starting with the lowest dominance depth level, the partitions are moved one by one to the new population as long as the first partition is reached that cannot be transfered completely. This corresponds to the scheme used in most hypervolume-based multi-objective optimizers (Emmerich et al., 2005; Igel et al., 2007; Brockhoff and Zitzler, 2007).

2. The partition that only fits partially into the new population is then processed using the method presented in Section 3.2. In each step, the fitness values for the partition under consideration are computed and the individual with the worst fitness is removed—if multiple individuals share the same minimal fitness, then one of them is selected uniformly at random. This procedure is repeated until the partition has been reduced to the desired size, that is, until it fits into the remaining slots left in the new population.

Concerning the fitness assignment, the number of objectives determines whether the exact or the estimated I_h^k values are considered. If less than four objectives are involved, we recommend employing Algorithm 1; otherwise use Algorithm 3. The latter works with a fixed number of sampling points to estimate the hypervolume

Algorithm 6 HypE Environmental Selection

```
Require: population $P \in \Psi$, reference set $R \subseteq Z$, number of offspring $N \in \mathbb{N}$, number
    of sampling points $M \in \mathbb{N}$
 1: procedure environmentalSelection($P, R, N, M$)
 2:     $P' \leftarrow P$                                  /*remaining population members*/
 3:     $Q \leftarrow \emptyset$                                          /*new population*/
 4:     $Q' \leftarrow \emptyset$                                 /*current nondominated set*/
 5:     repeat                             /*iteratively copy nondominated sets to $Q$*/
 6:         $Q \leftarrow Q \cup Q'$
 7:         $Q', P'' \leftarrow \emptyset$
 8:         for all $a \in P'$ do               /*determine current nondominated set in $P'$*/
 9:             if $\forall b \in P' : b \preceq a \Rightarrow a \preceq b$ then
10:                 $Q' \leftarrow Q' \cup \{a\}$
11:             else
12:                 $P'' \leftarrow P'' \cup \{a\}$
13:             end if
14:         end for
15:         $P' \leftarrow P''$
16:     until $|Q| + |Q'| \geq N \vee P' = \emptyset$
17:     $k = |Q| + |Q'| - N$
18:     while $k > 0$ do                   /*truncate last nonfitting nondominated set $Q'$*/
19:         if $n \leq 3$ then
20:             $\mathcal{F} \leftarrow computeHypervolume(Q', R, k)$
21:         else
22:             $\mathcal{F} \leftarrow estimateHypervolume(Q', R, k, M)$
23:         end if
24:         $Q' \leftarrow \emptyset$
25:         removed $\leftarrow$ false
26:         for all $(a, v) \in \mathcal{F}$ do                 /*remove worst solution from $Q'$*/
27:             if removed = true $\vee$ $v \neq \min_{(a,v)\in\mathcal{F}}\{v\}$ then
28:                 $Q' \leftarrow Q' \cup \{a\}$
29:             else
30:                 removed $\leftarrow$ true
31:             end if
32:         end for
33:         $k \leftarrow k - 1$
34:     end while
35:     $Q \leftarrow Q \cup Q'$
36:     return $Q$
37: end procedure
```

values I_h^k, regardless of the confidence of the decision to be made; hence, the variance of the estimates does not need to be calculated and it is sufficient to update for each sample drawn an array storing the fitness values of the population members.

6 Experiments

This section serves two goals: (i) to investigate the influence of specific algorithmic concepts (fitness, sample size) on the performance of HypE, and (ii) to study the effectiveness of HypE in comparison to existing MOEAs. A difficulty that arises in this context is how to statistically compare the quality of Pareto set approximations with respect to the hypervolume indicator when a large number of objectives ($n \geq 5$) is considered. In this case, exact computation of the hypervolume becomes infeasible; to this end, we propose Monte Carlo sampling using appropriate statistical tools as detailed below.

6.1 Experimental Setup

HypE is implemented within the PISA framework (Bleuler et al., 2003) and tested in two versions: the first (HypE) uses fitness-based mating selection as described in Algorithm 5, while the second (HypE*) employs a uniform mating selection scheme where all individuals have the same probability of being chosen for reproduction. Unless stated otherwise, for sampling the number of sampling points is fixed to $M =$ 10,000, kept constant during a run.

HypE and HypE* are compared to three popular MOEAs, namely NSGA-II (Deb et al., 2000), SPEA2 (Zitzler et al., 2002), and IBEA (in combination with the ϵ-indicator) (Zitzler and Künzli, 2004). Since these algorithms are not designed to optimize the hypervolume, it cannot be expected that they perform particularly well when measuring the quality of the approximation in terms of the hypervolume indicator. Nevertheless, they serve as an important reference as they are considerably faster than hypervolume-based search algorithms and therefore can execute a substantially larger number of generations when keeping the available computation time fixed. On the other hand, dedicated hypervolume-based methods are included in the comparisons. The algorithms proposed in previous works (Emmerich et al., 2005; Igel et al., 2007; Brockhoff and Zitzler, 2007) use the same fitness assignment scheme which can be mimicked by means of a HypE variant that only uses the I_h^1 values for fitness assignment, that is, k is set to 1, and employs the routine for exact hypervolume calculation (Algorithm 1). We will refer to this approach as RHV (regular hypervolume-based algorithm), where the acronym RHV* stands for the variant that uses uniform selection for mating. However, we do not provide comparisons to the original implementations of Emmerich et al. (2005), Igel et al. (2007), and Brockhoff and Zitzler (2007), because the focus is on the fitness assignment principles and not on specific data structures for fast hypervolume calculation as in Emmerich et al. (2005) or specific variation operators as in Igel et al. (2007). Furthermore, we consider the sampling-based optimizer proposed by Bader et al. (2010), here denoted as SHV (sampling-based hypervolume-oriented algorithm); it more or less corresponds to RHV with adaptive sampling. Finally, to study the influence of the nondominated sorting, we also include a simple HypE variant named RS (random selection) where all individuals are assigned the same constant fitness value. Thereby, the selection pressure is only maintained by the nondominated sorting carried out during the environmental selection phase.

As a basis for the comparisons, the DTLZ (Deb et al., 2005), the WFG (Huband et al., 2006), and the knapsack (Zitzler and Thiele, 1999) test problem suites are considered since they allow the number of objectives to be scaled arbitrarily (here, ranging from two to 50 objectives[6]). For the DTLZ problem, the number of decision variables is set

[6]Although problems with as many as 50 objectives exist, the objectives are thereby usually not genuinely independent, and hence their number can be reduced. Nevertheless, this study also includes as many as 50 objectives to demonstrate the potential of HypE.

Table 4: Number of decision variables and their decomposition into position and distance variables as used for the WFG test functions depending on the number of objectives.

	Objective space dimensions (n)						
	2d	3d	5d	7d	10d	25d	50d
Distance parameters	20	20	42	58	50	76	150
Position parameters	4	4	8	12	9	24	49
Decision variables	24	24	50	70	59	100	199

to 300, while for the WFG problems, individual values are used (see Table 4). As to the knapsack problem, we used 400 items that were modified with mutation probability 1 by one-bit mutation and by one-point crossover with probability 0.5. For each benchmark function, 30 runs are carried out per algorithm using a population size of $N = 50$ and a maximum number $g_{\max} = 200$ of generations (unless the computation time is fixed). The individuals are represented by real vectors, where a polynomial distribution is used for mutation and the SBX-20 operator for recombination (Deb, 2001). The recombination and mutation probabilities are set according to Deb et al. (2005).

6.2 Statistical Comparison Methodology

The quality of the Pareto set approximations are assessed using the hypervolume indicator, where for less than six objectives, the indicator values are calculated exactly, and otherwise approximated by Monte Carlo sampling as described in Bader and Zitzler (2008). When sampling is used, uncertainty of measurement is introduced, which can be expressed by the standard deviation of the sampled value $u(\hat{I}_H(A, R)) = I_H(A, R)\sqrt{p(1-p)/n}$, where p denotes the hit probability of the sampling process and $\hat{I}_H$ the hypervolume estimate. Unless otherwise noted, 1,000,000 samples are used per Pareto set approximation. For a typical hit probability between 10% to 90% observed, this leads to a very small uncertainty below 10^{-3} in relation to I_H. Therefore, it is highly unlikely that the uncertainty will influence the statistical test applied to the hypervolume estimates. If the uncertainty does exert an influence, then the statistical tests become overconservative. Hence, we do not consider uncertainty in the following tests.

Let A_i with $1 \leq i \leq l$ denote the algorithms to be compared. For each algorithm A_i, the same number r of independent runs are carried out for 200 generations. For formal reasons, the null hypothesis that all algorithms are equally well suited to approximate the Pareto-optimal set is investigated first, using the Kruskal-Wallis test at a significance level of $\alpha = 0.01$ (Conover, 1999). This hypothesis could be rejected in all test cases described below. Thereafter, for all pairs of algorithms, the difference in median of the hypervolume is compared.

In order to test the difference for significance, the Conover-Inman procedure is applied with the same α level as in the Kruskal-Wallis test (Conover, 1999). Let $\delta_{i,j}$ be 1 if A_i turns out to be significantly better than A_j, and 0 otherwise. Based on $\delta_{i,j}$, for each algorithm A_i the performance index $P(A_i)$ is determined as follows:

$$P(A_i) = \sum_{\substack{j=1 \\ j \neq i}}^{l} \delta_{i,j}. \tag{20}$$

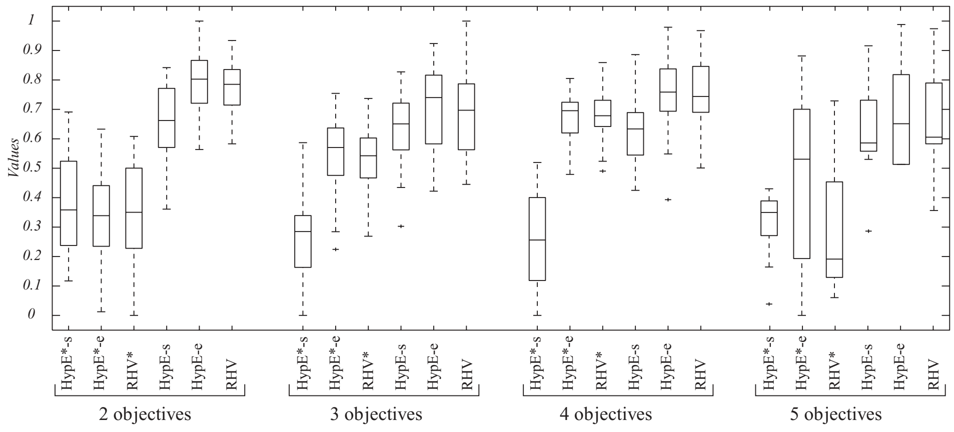


Figure 6: Comparison of the hypervolume indicator values for different variants of HypE and the regular hypervolume algorithm (RHV) on DTLZ2 with 2, 3, 4, and 5 objectives. For presentation reasons, the hypervolume values are normalized to the minimal and maximal values observed per problem instance.

This value reveals how many other algorithms are better than the corresponding algorithm on the specific test case. The smaller the index, the better the algorithm; an index of zero means that no other algorithm generated significantly better Pareto set approximations in terms of the hypervolume indicator.

6.3 Results

In the following, we discuss the experimental results grouped according to the foci of the investigations.

6.3.1 Exact Hypervolume Computation Versus Sampling

Next, we compare HypE with RHV. Due to the large computation effort caused by the exact hypervolume calculation, we use only one test problem, namely DTLZ2 with 2, 3, 4, and 5 objectives. Both HypE and HypE* are run with exact fitness calculation (Algorithm 1) as well as with the estimation procedure (Algorithm 3); the former variants are marked with a trailing "e," while the latter variants are marked with a trailing "s." All algorithms run for 200 generations and 30 runs per algorithm were performed.

Figure 6 shows the hypervolume values normalized for each test problem instance separately. As one might expect, HypE beats HypE*. Moreover, fitness-based mating selection is beneficial to both HypE and RHV. The two best variants, HypE-e and RHV, reach about the same hypervolume values independent of the number of objectives. Although HypE reaches a better hypervolume median for all four of the different numbers of objectives, the difference is never significant.[7] Hence, HypE can be considered an adequate alternative to the regular hypervolume algorithms; the main advantage though becomes evident when the respective fitness measures need to be estimated, as will be covered below.

[7]According to the Kruskal-Wallis test described in Section 6.2 with confidence level 0.01.

6.3.2 HypE Versus Other MOEAs

Now we compare HypE and HypE*, both using a constant number of samples, to other multi-objective EAs. Table 5 shows the performance score and mean hypervolume on the 17 test problems mentioned in Section 6.1. Except on a few test problems, HypE is better than HypE*. HypE reaches the best performance score overall. Summing up all performance scores, HypE yields the best total (76), followed by HypE* (143), IBEA (171), and the method proposed in Bader et al. (2010), which yields a total of 295. SPEA2 and NSGA-II reach almost the same score (413 and 421, respectively), clearly outperforming the random selection (626).

In order to better visualize the performance index, we show two figures where the index is summarized for different test problems and numbers of objectives. Figure 7 shows the average performance over all test problems for different number of objectives. Except for two objective problems, HypE yields the best score, increasing its lead in higher dimensions. The version using uniform mating selection, HypE*, is outperformed by IBEA for two to seven objectives and only thereafter reaches a similar score as HypE. This indicates that using nonuniform mating selection is particularly advantageous for a small number of objectives.

Next we look at the performance score for the individual test problems. Figure 8 shows the average index over all numbers of objectives. For DTLZ2, 4, 5, and 7, knapsack, and WFG8, IBEA outperforms HypE. For DTLZ7 and knapsack, SHV is better than HypE. For WFG4, HypE* has the lowest hypervolume. For the remaining 10 test problems, HypE reaches the best mean performance.

Note that the above comparison is carried out for the case of all algorithms run for the same number of generations, and HypE needs a longer execution time, for example, in comparison to SPEA2 or NSGA-II. We therefore investigate in the following, whether NSGA-II and SPEA2 will not overtake HypE given a constant amount of time. Figure 9 shows the hypervolume of the Pareto set approximations over time for HypE using the exact fitness values as well as the estimated values for different samples sizes M. Although only the results on WFG9 are shown, the same experiments were repeated on DTLZ2, DTLZ7, WFG3, and WFG6, and provided similar outcomes. Even though SPEA2, NSGA-II, and even IBEA are able to process twice as many generations as the exact HypE, they do not reach its hypervolume. In the three dimensional example used, HypE can be run sufficiently quickly without approximating the fitness values. Nevertheless, the sampled version is used as well to show the dependence of the execution time and quality on the number of samples M. Via M, the execution time of HypE can be traded off against the quality of the Pareto set approximation. The fewer number of samples are used, the more the behavior of HypE resembles random selection. On the other hand, by increasing M, the quality of exact calculation can be achieved, at the expense of increasing the execution time. For example, with $M = 1{,}000$, HypE is able to carry out nearly the same number of generations as SPEA2 or NSGA-II, but the Pareto set is just as good as when 100,000 samples are used, producing only $1/15$ the number of generations. In the example given, $M = 10{,}000$ represents the best compromise, but the number of samples should be increased in two cases: (i) the fitness evaluation takes more time, and (ii) more generations are used. Case (i) will affect the faster algorithm much more and increasing the number of samples will influence the execution time much less. Most real world problems, for instance, are considerably more expensive to evaluate than the DTLZ, WFG, and knapsack instances used in this paper. Therefore, the cost of the hypervolume estimation will matter less in most applications.

Table 5: Comparison of HypE to different MOEAs with respect to the hypervolume indicator. The first number represents the performance score P, which stands for the number of participants significantly dominating the selected algorithm. The number in parentheses denotes the hypervolume value, normalized to the minimum and maximum value observed on the test problem.

Problem	SHV	IBEA	NSGA-II	RS	SPEA2	HypE	HypE*
			Two objectives				
DTLZ 1	3 *(0.286)*	0 *(0.667)*	2 *(0.441)*	3 *(0.306)*	3 *(0.343)*	1 *(0.545)*	3 *(0.279)*
DTLZ 2	2 *(0.438)*	0 *(0.871)*	5 *(0.306)*	5 *(0.278)*	2 *(0.431)*	1 *(0.682)*	4 *(0.362)*
DTLZ 3	6 *(0.265)*	0 *(0.759)*	1 *(0.596)*	3 *(0.452)*	1 *(0.578)*	3 *(0.454)*	2 *(0.483)*
DTLZ 4	1 *(0.848)*	0 *(0.928)*	3 *(0.732)*	3 *(0.834)*	2 *(0.769)*	1 *(0.779)*	3 *(0.711)*
DTLZ 5	2 *(0.489)*	0 *(0.931)*	5 *(0.361)*	6 *(0.279)*	2 *(0.463)*	1 *(0.724)*	4 *(0.428)*
DTLZ 6	2 *(0.670)*	0 *(0.914)*	5 *(0.326)*	4 *(0.388)*	6 *(0.229)*	1 *(0.856)*	2 *(0.659)*
DTLZ 7	0 *(0.945)*	1 *(0.898)*	6 *(0.739)*	2 *(0.818)*	4 *(0.817)*	2 *(0.853)*	1 *(0.876)*
Knapsack	2 *(0.523)*	0 *(0.631)*	0 *(0.603)*	3 *(0.493)*	0 *(0.574)*	0 *(0.633)*	0 *(0.630)*
WFG 1	4 *(0.567)*	0 *(0.949)*	1 *(0.792)*	6 *(0.160)*	1 *(0.776)*	2 *(0.744)*	4 *(0.557)*
WFG 2	1 *(0.987)*	4 *(0.962)*	3 *(0.974)*	6 *(0.702)*	4 *(0.969)*	0 *(0.990)*	0 *(0.989)*
WFG 3	2 *(0.994)*	0 *(0.997)*	4 *(0.991)*	6 *(0.559)*	4 *(0.990)*	0 *(0.997)*	2 *(0.994)*
WFG 4	0 *(0.964)*	0 *(0.969)*	4 *(0.891)*	6 *(0.314)*	4 *(0.898)*	0 *(0.968)*	0 *(0.963)*
WFG 5	3 *(0.994)*	0 *(0.997)*	5 *(0.992)*	6 *(0.402)*	2 *(0.995)*	0 *(0.998)*	2 *(0.995)*
WFG 6	2 *(0.945)*	0 *(0.975)*	4 *(0.932)*	6 *(0.418)*	4 *(0.930)*	1 *(0.955)*	2 *(0.942)*
WFG 7	3 *(0.929)*	0 *(0.988)*	1 *(0.946)*	6 *(0.294)*	2 *(0.939)*	1 *(0.947)*	4 *(0.920)*
WFG 8	3 *(0.431)*	0 *(0.675)*	1 *(0.536)*	3 *(0.367)*	1 *(0.514)*	0 *(0.683)*	1 *(0.549)*
WFG 9	1 *(0.920)*	0 *(0.939)*	4 *(0.891)*	6 *(0.313)*	4 *(0.878)*	1 *(0.924)*	0 *(0.931)*
			Three objectives				
DTLZ 1	3 *(0.313)*	1 *(0.505)*	6 *(0.168)*	0 *(0.607)*	5 *(0.275)*	1 *(0.395)*	3 *(0.336)*
DTLZ 2	2 *(0.995)*	0 *(0.998)*	5 *(0.683)*	6 *(0.491)*	4 *(0.888)*	1 *(0.996)*	3 *(0.994)*
DTLZ 3	3 *(0.210)*	1 *(0.495)*	3 *(0.179)*	0 *(0.679)*	3 *(0.216)*	2 *(0.398)*	3 *(0.196)*
DTLZ 4	1 *(0.945)*	0 *(0.989)*	3 *(0.777)*	3 *(0.774)*	2 *(0.860)*	0 *(0.987)*	2 *(0.922)*
DTLZ 5	1 *(0.991)*	0 *(0.994)*	5 *(0.696)*	6 *(0.374)*	4 *(0.882)*	2 *(0.990)*	3 *(0.989)*
DTLZ 6	2 *(0.971)*	0 *(0.990)*	6 *(0.151)*	5 *(0.237)*	4 *(0.266)*	0 *(0.991)*	3 *(0.967)*
DTLZ 7	0 *(0.993)*	1 *(0.987)*	6 *(0.633)*	4 *(0.794)*	5 *(0.722)*	3 *(0.970)*	2 *(0.980)*
Knapsack	2 *(0.441)*	0 *(0.544)*	1 *(0.462)*	6 *(0.322)*	1 *(0.441)*	0 *(0.550)*	0 *(0.473)*
WFG 1	4 *(0.792)*	3 *(0.811)*	3 *(0.827)*	6 *(0.207)*	1 *(0.881)*	0 *(0.985)*	1 *(0.894)*
WFG 2	0 *(0.556)*	3 *(0.475)*	3 *(0.406)*	6 *(0.261)*	2 *(0.441)*	0 *(0.446)*	0 *(0.372)*
WFG 3	2 *(0.995)*	3 *(0.981)*	4 *(0.966)*	6 *(0.689)*	4 *(0.966)*	0 *(0.999)*	1 *(0.998)*
WFG 4	0 *(0.978)*	3 *(0.955)*	5 *(0.708)*	6 *(0.220)*	4 *(0.740)*	1 *(0.975)*	0 *(0.979)*
WFG 5	2 *(0.988)*	3 *(0.952)*	4 *(0.884)*	6 *(0.343)*	5 *(0.877)*	0 *(0.991)*	0 *(0.991)*
WFG 6	2 *(0.959)*	2 *(0.955)*	4 *(0.914)*	6 *(0.415)*	5 *(0.879)*	0 *(0.987)*	1 *(0.981)*
WFG 7	1 *(0.965)*	3 *(0.950)*	5 *(0.770)*	6 *(0.183)*	4 *(0.858)*	0 *(0.988)*	2 *(0.958)*
WFG 8	2 *(0.887)*	0 *(0.922)*	4 *(0.842)*	6 *(0.301)*	5 *(0.780)*	0 *(0.906)*	3 *(0.870)*
WFG 9	1 *(0.954)*	3 *(0.914)*	5 *(0.735)*	6 *(0.283)*	4 *(0.766)*	0 *(0.972)*	1 *(0.956)*
			Five objectives				
DTLZ 1	2 *(0.927)*	3 *(0.905)*	5 *(0.831)*	6 *(0.548)*	4 *(0.869)*	0 *(0.968)*	1 *(0.961)*
DTLZ 2	1 *(0.998)*	0 *(0.999)*	4 *(0.808)*	6 *(0.324)*	5 *(0.795)*	2 *(0.998)*	3 *(0.998)*
DTLZ 3	2 *(0.754)*	1 *(0.786)*	6 *(0.365)*	4 *(0.529)*	4 *(0.520)*	0 *(0.824)*	1 *(0.768)*
DTLZ 4	1 *(0.997)*	0 *(0.998)*	4 *(0.749)*	5 *(0.558)*	6 *(0.537)*	2 *(0.992)*	2 *(0.992)*
DTLZ 5	0 *(0.997)*	0 *(0.998)*	4 *(0.854)*	6 *(0.403)*	5 *(0.841)*	2 *(0.996)*	2 *(0.995)*
DTLZ 6	3 *(0.964)*	1 *(0.979)*	5 *(0.428)*	6 *(0.311)*	4 *(0.597)*	0 *(0.988)*	1 *(0.977)*
DTLZ 7	0 *(0.988)*	0 *(0.986)*	6 *(0.478)*	4 *(0.672)*	5 *(0.569)*	2 *(0.868)*	2 *(0.862)*
Knapsack	0 *(0.676)*	0 *(0.862)*	2 *(0.163)*	2 *(0.235)*	1 *(0.369)*	2 *(0.242)*	2 *(0.256)*

Table 5: Continued.

Problem	SHV	IBEA	NSGA-II	RS	SPEA2	HypE	HypE*
WFG 1	4 *(0.766)*	5 *(0.703)*	2 *(0.832)*	6 *(0.291)*	2 *(0.820)*	0 *(0.973)*	1 *(0.951)*
WFG 2	0 *(0.671)*	0 *(0.533)*	0 *(0.644)*	6 *(0.351)*	0 *(0.624)*	0 *(0.557)*	3 *(0.503)*
WFG 3	6 *(0.339)*	0 *(0.974)*	3 *(0.946)*	5 *(0.760)*	4 *(0.932)*	0 *(0.977)*	0 *(0.971)*
WFG 4	0 *(0.965)*	3 *(0.894)*	5 *(0.711)*	6 *(0.241)*	4 *(0.741)*	1 *(0.948)*	1 *(0.949)*
WFG 5	5 *(0.754)*	1 *(0.971)*	4 *(0.892)*	6 *(0.303)*	3 *(0.911)*	0 *(0.978)*	1 *(0.975)*
WFG 6	0 *(0.953)*	0 *(0.949)*	4 *(0.913)*	6 *(0.392)*	5 *(0.872)*	1 *(0.948)*	2 *(0.940)*
WFG 7	0 *(0.921)*	1 *(0.822)*	2 *(0.774)*	6 *(0.157)*	4 *(0.745)*	2 *(0.784)*	5 *(0.700)*
WFG 8	0 *(0.847)*	0 *(0.856)*	4 *(0.685)*	6 *(0.309)*	5 *(0.588)*	2 *(0.825)*	3 *(0.809)*
WFG 9	5 *(0.496)*	2 *(0.720)*	4 *(0.645)*	6 *(0.138)*	3 *(0.667)*	0 *(0.937)*	0 *(0.956)*
Seven objectives							
DTLZ 1	2 *(0.962)*	2 *(0.960)*	5 *(0.950)*	6 *(0.563)*	2 *(0.961)*	0 *(0.995)*	0 *(0.995)*
DTLZ 2	3 *(0.998)*	0 *(1.000)*	5 *(0.808)*	6 *(0.340)*	4 *(0.850)*	1 *(0.999)*	1 *(0.999)*
DTLZ 3	1 *(0.951)*	1 *(0.958)*	5 *(0.589)*	6 *(0.438)*	4 *(0.723)*	0 *(0.973)*	1 *(0.952)*
DTLZ 4	1 *(0.999)*	0 *(1.000)*	4 *(0.902)*	6 *(0.569)*	5 *(0.814)*	2 *(0.999)*	2 *(0.999)*
DTLZ 5	1 *(0.997)*	0 *(0.997)*	4 *(0.888)*	6 *(0.502)*	4 *(0.899)*	0 *(0.997)*	1 *(0.997)*
DTLZ 6	3 *(0.954)*	2 *(0.983)*	5 *(0.635)*	6 *(0.397)*	4 *(0.756)*	0 *(0.993)*	1 *(0.988)*
DTLZ 7	0 *(0.981)*	1 *(0.958)*	5 *(0.348)*	4 *(0.559)*	5 *(0.352)*	2 *(0.877)*	2 *(0.870)*
Knapsack	0 *(0.745)*	0 *(0.768)*	2 *(0.235)*	2 *(0.226)*	2 *(0.272)*	2 *(0.276)*	4 *(0.212)*
WFG 1	4 *(0.647)*	5 *(0.649)*	2 *(0.814)*	6 *(0.189)*	2 *(0.812)*	0 *(0.956)*	1 *(0.937)*
WFG 2	0 *(0.632)*	0 *(0.747)*	1 *(0.409)*	5 *(0.155)*	0 *(0.837)*	0 *(0.528)*	0 *(0.630)*
WFG 3	6 *(0.105)*	2 *(0.975)*	3 *(0.961)*	5 *(0.709)*	4 *(0.958)*	0 *(0.983)*	0 *(0.982)*
WFG 4	3 *(0.888)*	2 *(0.919)*	4 *(0.688)*	6 *(0.200)*	4 *(0.694)*	0 *(0.956)*	0 *(0.952)*
WFG 5	6 *(0.042)*	2 *(0.982)*	4 *(0.905)*	5 *(0.406)*	3 *(0.938)*	0 *(0.986)*	0 *(0.987)*
WFG 6	0 *(0.978)*	0 *(0.967)*	4 *(0.940)*	6 *(0.453)*	5 *(0.921)*	0 *(0.974)*	3 *(0.967)*
WFG 7	1 *(0.688)*	3 *(0.657)*	0 *(0.813)*	6 *(0.207)*	3 *(0.658)*	1 *(0.713)*	5 *(0.606)*
WFG 8	0 *(0.933)*	1 *(0.905)*	4 *(0.709)*	6 *(0.366)*	5 *(0.537)*	2 *(0.863)*	2 *(0.874)*
WFG 9	5 *(0.385)*	2 *(0.681)*	3 *(0.679)*	6 *(0.119)*	3 *(0.683)*	0 *(0.928)*	0 *(0.943)*
Ten objectives							
DTLZ 1	3 *(0.981)*	5 *(0.971)*	4 *(0.986)*	6 *(0.590)*	2 *(0.990)*	0 *(0.999)*	0 *(0.999)*
DTLZ 2	3 *(0.999)*	2 *(1.000)*	5 *(0.825)*	6 *(0.290)*	4 *(0.868)*	0 *(1.000)*	0 *(1.000)*
DTLZ 3	3 *(0.951)*	1 *(0.990)*	5 *(0.676)*	6 *(0.358)*	4 *(0.750)*	0 *(0.994)*	1 *(0.990)*
DTLZ 4	2 *(1.000)*	0 *(1.000)*	4 *(0.988)*	6 *(0.560)*	5 *(0.960)*	1 *(1.000)*	0 *(1.000)*
DTLZ 5	3 *(0.951)*	0 *(0.998)*	4 *(0.899)*	6 *(0.471)*	4 *(0.892)*	0 *(0.998)*	1 *(0.997)*
DTLZ 6	4 *(0.497)*	2 *(0.987)*	4 *(0.706)*	6 *(0.276)*	3 *(0.769)*	0 *(0.994)*	1 *(0.992)*
DTLZ 7	0 *(0.986)*	1 *(0.831)*	4 *(0.137)*	6 *(0.057)*	4 *(0.166)*	2 *(0.744)*	1 *(0.781)*
Knapsack	0 *(0.568)*	0 *(0.529)*	2 *(0.149)*	4 *(0.119)*	2 *(0.173)*	5 *(0.068)*	5 *(0.060)*
WFG 1	6 *(0.402)*	4 *(0.843)*	2 *(0.932)*	5 *(0.562)*	2 *(0.937)*	0 *(0.977)*	0 *(0.975)*
WFG 2	0 *(0.971)*	0 *(0.988)*	0 *(0.978)*	5 *(0.020)*	2 *(0.962)*	0 *(0.981)*	1 *(0.966)*
WFG 3	6 *(0.088)*	1 *(0.973)*	3 *(0.947)*	5 *(0.792)*	4 *(0.933)*	0 *(0.980)*	1 *(0.976)*
WFG 4	3 *(0.698)*	2 *(0.896)*	3 *(0.708)*	6 *(0.207)*	5 *(0.669)*	0 *(0.950)*	0 *(0.955)*
WFG 5	6 *(0.014)*	2 *(0.979)*	4 *(0.832)*	5 *(0.365)*	3 *(0.913)*	0 *(0.987)*	0 *(0.989)*
WFG 6	3 *(0.934)*	1 *(0.949)*	4 *(0.896)*	6 *(0.449)*	5 *(0.865)*	0 *(0.959)*	1 *(0.949)*
WFG 7	1 *(0.686)*	4 *(0.464)*	1 *(0.604)*	6 *(0.077)*	4 *(0.473)*	0 *(0.683)*	3 *(0.548)*
WFG 8	0 *(0.956)*	1 *(0.903)*	4 *(0.689)*	6 *(0.221)*	5 *(0.438)*	2 *(0.883)*	2 *(0.875)*
WFG 9	5 *(0.222)*	3 *(0.584)*	3 *(0.644)*	6 *(0.109)*	2 *(0.676)*	1 *(0.893)*	0 *(0.925)*

Table 5: Continued.

Problem	SHV	IBEA	NSGA-II	RS	SPEA2	HypE	HypE*
			Twenty-five objectives				
DTLZ 1	4 *(0.994)*	5 *(0.987)*	2 *(1.000)*	6 *(0.657)*	3 *(0.998)*	0 *(1.000)*	0 *(1.000)*
DTLZ 2	3 *(0.999)*	2 *(1.000)*	4 *(0.965)*	6 *(0.301)*	5 *(0.882)*	0 *(1.000)*	0 *(1.000)*
DTLZ 3	3 *(0.967)*	2 *(0.999)*	4 *(0.930)*	6 *(0.455)*	5 *(0.827)*	0 *(0.999)*	0 *(1.000)*
DTLZ 4	3 *(1.000)*	2 *(1.000)*	4 *(1.000)*	6 *(0.546)*	5 *(0.991)*	0 *(1.000)*	0 *(1.000)*
DTLZ 5	5 *(0.781)*	2 *(0.996)*	3 *(0.949)*	6 *(0.457)*	4 *(0.808)*	0 *(0.999)*	1 *(0.999)*
DTLZ 6	6 *(0.286)*	2 *(0.993)*	3 *(0.957)*	5 *(0.412)*	4 *(0.830)*	0 *(0.999)*	0 *(0.998)*
DTLZ 7	0 *(0.973)*	0 *(0.966)*	3 *(0.856)*	2 *(0.893)*	4 *(0.671)*	2 *(0.889)*	3 *(0.825)*
Knapsack	0 *(0.000)*	4 *(0.000)*	5 *(0.000)*	3 *(0.000)*	6 *(0.000)*	1 *(0.000)*	2 *(0.000)*
WFG 1	6 *(0.183)*	4 *(0.930)*	0 *(0.971)*	5 *(0.815)*	3 *(0.965)*	0 *(0.972)*	0 *(0.973)*
WFG 2	0 *(0.951)*	0 *(0.951)*	2 *(0.935)*	6 *(0.072)*	2 *(0.933)*	2 *(0.934)*	2 *(0.928)*
WFG 3	6 *(0.037)*	0 *(0.983)*	3 *(0.965)*	5 *(0.758)*	3 *(0.963)*	1 *(0.974)*	1 *(0.977)*
WFG 4	6 *(0.063)*	2 *(0.890)*	3 *(0.541)*	5 *(0.170)*	4 *(0.432)*	0 *(0.941)*	0 *(0.945)*
WFG 5	6 *(0.003)*	3 *(0.832)*	4 *(0.796)*	5 *(0.227)*	2 *(0.915)*	0 *(0.989)*	0 *(0.989)*
WFG 6	3 *(0.932)*	0 *(0.959)*	5 *(0.913)*	6 *(0.579)*	3 *(0.926)*	0 *(0.961)*	0 *(0.962)*
WFG 7	3 *(0.286)*	4 *(0.183)*	2 *(0.386)*	6 *(0.081)*	4 *(0.185)*	0 *(0.707)*	1 *(0.479)*
WFG 8	0 *(0.924)*	0 *(0.909)*	4 *(0.517)*	6 *(0.189)*	5 *(0.305)*	2 *(0.817)*	3 *(0.792)*
WFG 9	5 *(0.118)*	3 *(0.531)*	3 *(0.580)*	5 *(0.133)*	2 *(0.681)*	0 *(0.893)*	1 *(0.848)*
			Fifty objectives				
DTLZ 1	4 *(0.992)*	5 *(0.985)*	2 *(1.000)*	6 *(0.566)*	3 *(0.999)*	1 *(1.000)*	0 *(1.000)*
DTLZ 2	3 *(1.000)*	2 *(1.000)*	4 *(0.998)*	6 *(0.375)*	5 *(0.917)*	0 *(1.000)*	0 *(1.000)*
DTLZ 3	3 *(0.984)*	2 *(1.000)*	3 *(0.988)*	6 *(0.518)*	5 *(0.891)*	0 *(1.000)*	0 *(1.000)*
DTLZ 4	2 *(1.000)*	2 *(1.000)*	4 *(1.000)*	6 *(0.517)*	5 *(0.999)*	0 *(1.000)*	0 *(1.000)*
DTLZ 5	5 *(0.477)*	2 *(0.996)*	3 *(0.954)*	5 *(0.425)*	4 *(0.752)*	0 *(0.999)*	0 *(0.999)*
DTLZ 6	6 *(0.112)*	2 *(0.995)*	3 *(0.979)*	5 *(0.399)*	4 *(0.839)*	0 *(0.998)*	1 *(0.998)*
DTLZ 7	1 *(0.767)*	0 *(0.966)*	5 *(0.233)*	4 *(0.254)*	6 *(0.020)*	2 *(0.684)*	3 *(0.675)*
Knapsack	0 *(0.000)*	4 *(0.000)*	5 *(0.000)*	3 *(0.000)*	6 *(0.000)*	1 *(0.000)*	2 *(0.000)*
WFG 1	6 *(0.210)*	4 *(0.869)*	2 *(0.962)*	4 *(0.823)*	2 *(0.961)*	0 *(0.971)*	0 *(0.970)*
WFG 2	3 *(0.538)*	0 *(0.962)*	0 *(0.959)*	6 *(0.076)*	0 *(0.952)*	2 *(0.945)*	3 *(0.943)*
WFG 3	6 *(0.059)*	0 *(0.981)*	2 *(0.972)*	5 *(0.731)*	2 *(0.973)*	0 *(0.976)*	0 *(0.979)*
WFG 4	6 *(0.011)*	2 *(0.783)*	3 *(0.268)*	5 *(0.118)*	3 *(0.258)*	0 *(0.944)*	1 *(0.908)*
WFG 5	6 *(0.003)*	2 *(0.940)*	4 *(0.789)*	5 *(0.416)*	3 *(0.913)*	1 *(0.987)*	0 *(0.989)*
WFG 6	4 *(0.933)*	2 *(0.963)*	4 *(0.941)*	6 *(0.663)*	2 *(0.961)*	0 *(0.974)*	0 *(0.976)*
WFG 7	1 *(0.312)*	5 *(0.026)*	3 *(0.208)*	5 *(0.022)*	4 *(0.034)*	0 *(0.581)*	1 *(0.378)*
WFG 8	1 *(0.669)*	0 *(0.913)*	4 *(0.341)*	6 *(0.147)*	5 *(0.233)*	1 *(0.602)*	2 *(0.579)*
WFG 9	5 *(0.250)*	3 *(0.597)*	3 *(0.559)*	6 *(0.166)*	2 *(0.727)*	0 *(0.907)*	0 *(0.903)*

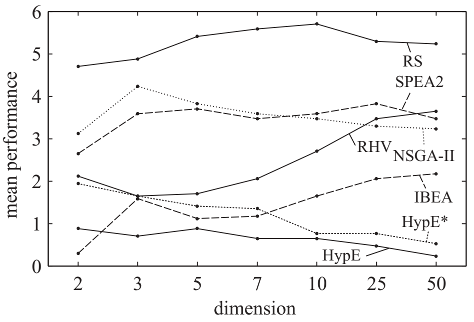


Figure 7: Mean performance score over all test problems for different number of objectives. The smaller the score, the better the Pareto set approximation in terms of hypervolume.

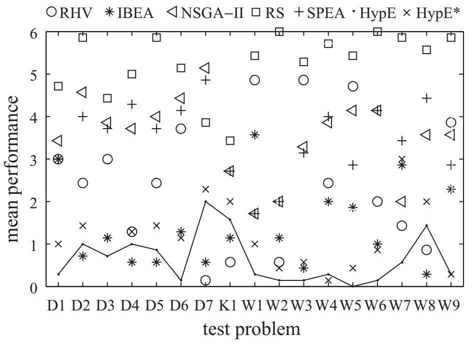


Figure 8: Mean performance score over all dimensions for different test problems, namely DTLZ (Dx), WFG (Wx), and knapsack (K1). The values of HypE+ are connected by a solid line to easier assess the score.

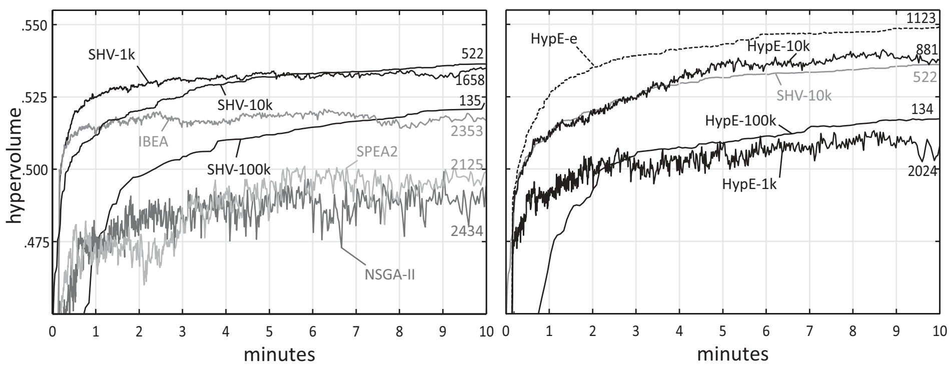


Figure 9: Hypervolume process over ten min of HypE+ for different sample sizes x in thousands (Hy-xk) as well as using the exact values (Hy-e). The test problem is WFG9 for three objectives. HypE is compared to the algorithms presented in Section 6, where the results are split into two figures with an identical axis for the sake of clarity. The numbers at the left border indicate the total number of generations.

In case (ii), HypE using more samples might overtake the faster versions with fewer samples, since those are more vulnerable to stagnation.

7 Conclusions

This paper proposes HypE, a novel hypervolume-based multi-objective evolutionary algorithm that can be applied to problems with arbitrary numbers of objective functions. It incorporates a new fitness assignment scheme based on the Lebesgue measure, where this measure can be both exactly calculated and estimated by means of Monte Carlo sampling. The latter allows for a trade-off of fitness accuracy against the overall computation time budget, which renders hypervolume-based search possible also for many-objective problems, in contrast to other reported algorithms (Emmerich et al., 2005; Igel et al., 2007; Brockhoff and Zitzler, 2007). HypE is available for download at http://www.tik.ee.ethz.ch/sop/download/supplementary/hype/.

HypE was compared to various state of the art MOEAs with regard to the hypervolume indicator values of the generated Pareto set approximations—on the DTLZ (Deb et al., 2005), the WFG (Huband et al., 2006), and the knapsack (Zitzler and Thiele, 1999) test problem suites. The simulations results indicate that HypE is a highly competitive multi-objective search algorithm; in the considered setting, the Pareto front approximations obtained by HypE reached the best hypervolume value in six out of seven cases averaged over all test problems.

A promising direction of future research is the development of advanced adaptive sampling strategies that exploit the available computing resources most effectively, such as increasing the number of samples toward the end of the evolutionary run.

Acknowledgments

Johannes Bader has been supported by the Indo-Swiss Joint Research Program under grant IT14.

References

Bader, J., Deb, K., and Zitzler, E. (2010). Faster hypervolume-based search using Monte Carlo sampling. In M. Ehrgott et al. (Eds.), *Conference on Multiple Criteria Decision Making (MCDM 2008)*, Vol. 634 of *LNEMS*, pp. 313–326.

Bader, J., and Zitzler, E. (2008). HypE: An algorithm for fast hypervolume-based many-objective optimization. TIK Report 286, Computer Engineering and Networks Laboratory (TIK), ETH Zurich.

Beume, N., Fonseca, C. M., Lopez-Ibanez, M., Paquete, L., and Vahrenhold, J. (2007). On the complexity of computing the hypervolume indicator. Tech. Rep. CI-235/07, University of Dortmund.

Beume, N., Naujoks, B., and Emmerich, M. (2007). SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal on Operational Research*, 181:1653–1669.

Beume, N., and Rudolph, G. (2006). Faster S-metric calculation by considering dominated hypervolume as Klee's measure problem. Tech. Rep. CI-216/06, Sonderforschungsbereich 531 Computational Intelligence, Universität Dortmund. shorter version published at IASTED International Conference on Computational Intelligence (CI 2006).

Bleuler, S., Laumanns, M., Thiele, L., and Zitzler, E. (2003). PISA—A platform and programming language independent interface for search algorithms. In C. M. Fonseca et al. (Eds.), *Conference on Evolutionary Multi-Criterion Optimization (EMO 2003)*, Vol. 2632 of *LNCS*, pp. 494–508. Berlin: Springer.

Bradstreet, L., and Barone, L. W. (2009). Updating exclusive hypervolume contributions cheaply. In *Congress on Evolutionary Computation (CEC'2009)*, pp. 538–544.

Bradstreet, L., Barone, L., and While, L. (2006). Maximising hypervolume for selection in multi-objective evolutionary algorithms. In *Congress on Evolutionary Computation (CEC 2006)*, pp. 6208–6215.

Bradstreet, L., While, L., and Barone, L. (2007). Incrementally maximising hypervolume for selection in multi-objective evolutionary algorithms. In *Congress on Evolutionary Computation (CEC 2007)*, pp. 3203–3210.

Bringmann, K., and Friedrich, T. (2008). Approximating the volume of unions and intersections of high-dimensional geometric objects. In S. H. Hong, H. Nagamochi, and T. Fukunaga (Eds.), *International Symposium on Algorithms and Computation (ISAAC 2008)*, Vol. 5369 of *LNCS*, pp. 436–447. Berlin: Springer.

Brockhoff, D., and Zitzler, E. (2007). Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods. In *Congress on Evolutionary Computation (CEC 2007)*, pp. 2086–2093.

Conover, W. J. (1999). *Practical nonparametric statistics* (3rd ed.). New York: John Wiley.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. New York: John Wiley.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In M. Schoenauer et al. (Eds.), *Conference on Parallel Problem Solving from Nature (PPSN VI)*, vol. 1917 of *LNCS*, pp. 849–858. Berlin: Springer.

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2005). Scalable test problems for evolutionary multi-objective optimization. In A. Abraham, R. Jain, and R. Goldberg (Eds.), *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, chap. 6 (pp. 105–145). Berlin: Springer.

Emmerich, M., Beume, N., and Naujoks, B. (2005). An EMO algorithm using the hypervolume measure as selection criterion. In *Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, Vol. 3410 of *LNCS*, pp. 62–76. Berlin: Springer.

Emmerich, M., Deutz, A., and Beume, N. (2007). Gradient-based/evolutionary relay hybrid for computing Pareto front approximations maximizing the S-metric. In *Hybrid Metaheuristics*, pp. 140–156. Berlin: Springer.

Everson, R., Fieldsend, J., and Singh, S. (2002). Full elite-sets for multiobjective optimisation. In I. Parmee (Eds.), *Conference on Adaptive Computing in Design and Manufacture (ADCM 2002)*, pp. 343–354.

Fleischer, M. (2003). The measure of Pareto optima. Applications to multi-objective metaheuristics. In C. M. Fonseca et al. (Eds.), *Conference on Evolutionary Multi-Criterion Optimization (EMO 2003)*, Vol. 2632 of *LNCS*, pp. 519–533. Berlin: Springer.

Fonseca, C. M., Paquete, L., and López-Ibáñez, M. (2006). An improved dimension-sweep algorithm for the hypervolume indicator. In *Congress on Evolutionary Computation (CEC 2006)*, pp. 1157–1163.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley.

Huband, S., Hingston, P., Barone, L., and While, L. (2006). A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506.

Huband, S., Hingston, P., White, L., and Barone, L. (2003). An evolution strategy with probabilistic mutation for multi-objective optimisation. In *Congress on Evolutionary Computation (CEC 2003)*, Vol. 3, pp. 2284–2291.

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

Knowles, J. (2002). *Local-search and hybrid evolutionary algorithms for Pareto optimization*. PhD thesis, University of Reading.

Knowles, J., and Corne, D. (2002). On metrics for comparing non-dominated sets. In *Congress on Evolutionary Computation (CEC 2002)*, pp. 711–716.

Knowles, J., and Corne, D. (2003). Properties of an adaptive archiving algorithm for storing nondominated vectors. *IEEE Transactions on Evolutionary Computation*, 7(2):100–116.

Laumanns, M., Rudolph, G., and Schwefel, H.-P. (1999). Approximating the Pareto set: Concepts, diversity issues, and performance assessment. Tech. Rep. CI-7299, University of Dortmund.

Mostaghim, S., Branke, J., and Schmeck, H. (2007). Multi-objective particle swarm optimization on computer grids. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO 2007)*, pp. 869–875.

Nicolini, M. (2005). A two-level evolutionary approach to multi-criterion optimization of water supply systems. In *Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, Vol. 3410 of *LNCS*, pp. 736–751. Berlin: Springer.

Obayashi, S., Deb, K., Poloni, C., Hiroyasu, T., and Murata, T. (Eds.) (2007). *Conference on Evolutionary Multi-Criterion Optimization (EMO 2007)*, Vol. 4403 of *LNCS*. Berlin: Springer.

Scharnow, J., Tinnefeld, K., and Wegener, I. (2004). The analysis of evolutionary algorithms on sorting and shortest paths problems. *Journal of Mathematical Modelling and Algorithms*, 3(4):349–366.

Van Veldhuizen, D. A. (1999). *Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations*. PhD thesis, Graduate School of Engineering, Air Force Institute of Technology, Air University.

Wagner, T., Beume, N., and Naujoks, B. (2007). Pareto-, aggregation-, and indicator-based methods in many-objective optimization. In S. Obayashi et al. (Eds.), *Conference on Evolutionary Multi-Criterion Optimization (EMO 2007)*, Vol. 4403 of *LNCS*, pp. 742–756. Berlin: Springer. Extended version published as internal report of Sonderforschungsbereich 531 Computational Intelligence CI-217/06, Universität Dortmund, September 2006.

While, L. (2005). A new analysis of the LebMeasure algorithm for calculating hypervolume. In *Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, Vol. 3410 of *LNCS*, pp. 326–340. Berlin: Springer.

While, L., Bradstreet, L., Barone, L., and Hingston, P. (2005). Heuristics for optimising the calculation of hypervolume for multi-objective optimisation problems. In *Congress on Evolutionary Computation (CEC 2005)*, pp. 2225–2232.

While, L., Hingston, P., Barone, L., and Huband, S. (2006). A faster algorithm for calculating hypervolume. *IEEE Transactions on Evolutionary Computation*, 10(1):29–38.

Yang, Q., and Ding, S. (2007). Novel algorithm to calculate hypervolume indicator of Pareto approximation set. In *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues, Third International Conference on Intelligent Computing (ICIC 2007)*, Vol. 2, pp. 235–244.

Zitzler, E. (1999). *Evolutionary algorithms for multiobjective optimization: Methods and applications*. PhD thesis, ETH Zurich, Switzerland.

Zitzler, E. (2001). Hypervolume metric calculation. ftp://ftp.tik.ee.ethz.ch/pub/people/zitzler/hypervol.c.

Zitzler, E., Brockhoff, D., and Thiele, L. (2007). The hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration. In S. Obayashi et al. (Eds.), *Conference on Evolutionary Multi-Criterion Optimization (EMO 2007)*, Vol. 4403 of *LNCS*, pp. 862–876, Berlin: Springer.

Zitzler, E., and Künzli, S. (2004). Indicator-based selection in multiobjective search. In X. Yao et al. (Eds.), *Conference on Parallel Problem Solving from Nature (PPSN VIII)*, Vol. 3242 of *LNCS*, pp. 832–842. Berlin: Springer.

Zitzler, E., Laumanns, M., and Thiele, L. (2002). SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In K. Giannakoglou et al. (Eds.), *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pp. 95–100. International Center for Numerical Methods in Engineering (CIMNE).

Zitzler, E., and Thiele, L. (1998a). An evolutionary approach for multiobjective optimization: The strength pareto approach. TIK Report 43, Computer Engineering and Networks Laboratory (TIK), ETH Zurich.

Zitzler, E., and Thiele, L. (1998b). Multiobjective optimization using evolutionary algorithms—A comparative case study. In *Conference on Parallel Problem Solving from Nature (PPSN V)*, pp. 292–301.

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

Zitzler, E., Thiele, L., and Bader, J. (2008). On set-based multiobjective optimization. TIK Report 300, Computer Engineering and Networks Laboratory (TIK), ETH Zurich.

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