# Self-adaptive Evolutionary Many-Objective Optimization Based on Relation $\varepsilon$-Preferred

**Nicole Drechsler**

**Abstract** Many real-world optimization problems consist of several mutually dependent subproblems. If more than three optimization objectives are involved in the optimization process, the so-called *Many-Objective Optimization* is a challenge in the area of multi-objective optimization. Often, the objectives have different levels of importance that have to be considered. For this, relation *$\varepsilon$-Preferred* has been presented, that enables to compare and rank multi-dimensional solutions. *$\varepsilon$-Preferred* is controlled by a parameter $\varepsilon$ that has influence on the quality of the results. In this paper for the setting of the epsilon values three heuristics have been investigated. To demonstrate the behavior and efficiency of these methods an *Evolutionary Algorithm* for the multi-dimensional *Nurse Rostering Problem* is proposed. It is shown by experiments that former approaches are outperformed by heuristics that are based on self-adaptive mechanisms.

**Keywords** Many-objective optimization · Nurse rostering problem · Relation $\varepsilon$-preferred · User preferences

## 1 Introduction

During the last 20 years solving *Multi-Objective Optimization* (MOO) problems is getting more and more important. Many real-world problems consist of multiple competing subproblems that have to be optimized in parallel. For *Evolutionary Algorithms* (EAs) many approaches have been presented that cope with MOO problems [1–4]. If more than three objectives are involved in the optimization process the corresponding problems are called many-objective optimization problems. Especially, if real-world optimization problems are of interest, more than three objectives are considered during the optimization process [5–7]. Furthermore, in industrial problems optimization criteria have often different levels of importance. These user

N. Drechsler (✉)
Institute of Computer Science, University of Bremen, 28359 Bremen, Germany
e-mail: nd@informatik.uni-bremen.de

© Springer International Publishing Switzerland 2016

K. Madani et al. (eds.), *Computational Intelligence*,
Studies in Computational Intelligence 613,
DOI 10.1007/978-3-319-23392-5_2

preferences have to be taken into account during optimization. Considering both, many-objective optimization and user preferences, there is a need for optimization models that combine these properties [8–12].

To overcome these problems one classical method to combine multiple optimization criteria with user preferences is the *Weighted Sum* approach. Here, a single value is computed by a linear combination of the considered criteria. By the choice of the weights for each criterion the influence of the user preference can be controlled. It is often used in industrial applications, because it is easy to implement and at a first view scales well. Further examinations have shown that it is a challenge to adjust the weights such that the search is guided in the desired direction [5, 13]. A disadvantage of the weighted sum approach is that it is incapable to find compromise solutions of concave *Pareto* fronts. A further classical approach is the use of *Non-dominated Sets* that is based on the *Pareto-Dominance* relation [14]. Using the *Dominance* relation a ranking between multi-dimensional solutions can be required. If EAs are used for MOO, the method NSGA-II [3] is a basic approach that is based on non-dominated sorting. It is suitable in low dimensions, but for more than three objectives more sophisticated approaches are required [15, 16]. As an alternative, the hypervolume indicator is proposed, i.e. to each candidate solution an indicator value is assigned, but due to the computational complexity it can only be applied in low dimensions. An approximation of the hypervolume for higher dimensions is presented in [4].

In further developments relation $\varepsilon$*-Preferred* has been proposed for many-objective optimization [17]. Using relation $\varepsilon$*-Preferred* a ranking between solutions can be determined and solutions that are incomparable using *Dominates* can be distinguished. Thus $\varepsilon$*-Preferred* is a refinement of relation *Dominates*. In [12] the model based on $\varepsilon$*-Preferred* has been enlarged such that it can also handle user preferences (priorities). For this model, the influence of parameter $\varepsilon$ has been investigated and method AEP, that determines $\varepsilon$ automatically, has been presented. In this context the *Nurse Rostering Problem* (NRP) has been considered, i.e. a scheduling problem where a working plan for employees in a hospital has to be computed. The proposed method is compared to the well-known NSGA-II approach, because the modeling of user preferences as proposed in this article can easily be used within this method. A comparison to more sophisticated approaches, like e.g. the MOEA/D [18] or the hypervolume approach [4] can not directly be performed: Taking the user defined priorities into account, the comparison to these approaches without user preference modeling is not meaningful, because the usage of user preferences is not provided.

In this paper an *Evolutionary Algorithm* that makes use of the $\varepsilon$*-Preferred* relation including user preferences is applied to the NRP.[1] In contrast to [12] the full potential of the presented model has been exploited. To model the user preferences in [12] only two types of priorities are used, the soft constraints and hard constraints. The hard constraints have a higher priority during optimization than the soft constraints. The hard constraints map the rules of the nurse station. Each soft constraint itself

[1]For the investigation of this approach the NRP has been used as application, because it consists of many objectives with different levels of priorities. There, in contrast to standard benchmarks for MOO (DTLZ [19]), the priorities are provided in the benchmark files.

consists of up to 90 rules that have different user preferences. These user preferences are directly given as weights in the benchmark examples [20]. Then, a weighted sum is constructed to compute the constraints. Using the soft and hard constraints, a multi-dimensional fitness function is computed, such that the hard constraint and for each employee the corresponding soft constraints are provided. Following this, benchmarks with up to 17 optimization criteria are considered. In contrast, in this approach each rule of the soft constraints is treated as a separated optimization criterion. This leads to fitness functions with up to 90 objectives. For each rule a priority is calculated dependent on the weight that is specified in the benchmark. In the experiments it is shown that the results from [12] can be further improved if the advanced model as described above is used.

Furthermore, the justification of the epsilon values is examined. Two self-adapting methods for epsilon adaptation are presented. It is shown in our experiments that AEP [12] can be further improved. Additionally, an approach based on *Weighted Sums* is outperformed, where previously published methods fail.

## 2 Preliminaries

First, we give a short introduction into the basic techniques of multi-objective optimization and relations used for comparison.

### *2.1 Multi-objective Optimization*

A multi-objective optimization problem is defined as follows: Given a search space $\boldsymbol{\Omega}$, an evaluation function $f : \boldsymbol{\Omega} \rightarrow \mathbb{R}^m$ is defined to calculate the fitness vector $F(A) : \forall A \in \boldsymbol{\Omega}$ of size $m$. Then we have to minimize (or maximize) the elements of $F(A)$. In the following we assume, without loss of generality, that $F$ has to be minimized for all objectives. According to [14] it holds:

**Definition 1** Let $A, B \in \boldsymbol{\Omega}$.

$$A \prec_{dominates} B :\Leftrightarrow \exists j : F_j(A) < F_j(B) : F_i(A) \leqslant F_i(B), 1 \leqslant i \leqslant m. \quad (1)$$

Based on this, we can describe the `Pareto set` as

$$\chi : \forall p \in \chi : \nexists q \in \boldsymbol{\Omega} : q \prec_{dominates} p. \quad (2)$$

It can be directly seen from the definition that for $A, B \in \boldsymbol{\Omega}$ element A dominates B only if A is better than B in at least one component and equal or better in all components. Relation *Dominates* is a partial order. In evolutionary multi-objective optimization relation *Dominates* is used to perform *Non-dominated Sorting* [3]: All elements of a population are compared using *dominates* and the non-dominated elements are computed. This set is called *Non-dominated Set*. Disregarding the *Non-dominated Set* the next level of non-dominated elements is considered. This is repeated, until all elements are classified. The elements $A \in \boldsymbol{\Omega}$ in the *Non-dominated Set* are equal or not comparable and hence, the designer is interested in solutions from the *Non-dominated Set*.

## 2.2 Relation Preferred

In [5] a refinement of relation *Dominates* has been presented. The approach is well-suited for problems in many-objective optimization, i.e. if more than three optimization criteria are considered. In [3] it has been shown that in higher dimensions more than 90 % of the population are *Non-dominated* elements and thus, a ranking used for selection mechanisms cannot be performed. To overcome these problems relation *Preferred* is defined as follows:

**Definition 2** Let $A, B \in \boldsymbol{\Omega}$ and $1 \leq i, j, \leq m$.

$$A \prec_{preferred} B :\Leftrightarrow |\{i : F_i(A) < F_i(B)\}| > |\{j : F_j(B) < F_j(A)\}|. \tag{3}$$

Relation *Preferred* considers the number of different objectives of A and B. A is *preferred* to B if $i\,(i < m)$ objectives of A are smaller or equal than the corresponding objectives in B and only $j\,(j < i)$ objectives of B are smaller or equal than the corresponding objectives in A.

Using relation *Preferred* the solutions in a population are classified in so-called *Satisfiability Classes* (SCs) [5]. All solutions $A \in \boldsymbol{\Omega}$ are compared using relation *Preferred*. Then the relation graph is constructed, where each element is a node and preferences are represented by edges. *Preferred* is not a partial order, because the relation graph can have cycles, and thus it is not transitive.

To overcome this property the relation graph is modified such that cycles are eliminated. The main idea is that elements that are included in a cycle should be ranked equally. For this the *Strongly Connected Components* (SCC) of the graph are computed by a linear time DFS-based algorithm [21]. Then the relation graph is modified such that each SCC is replaced by a new node representing all elements in the corresponding cycle. Doing so, all cycles in the relation are eliminated. The relation that is represented by the acyclic relation graph is transitive and antisymmetric, which is sufficient for our purposes. Level sorting of the nodes in the acyclic relation graph determines a ranking of SCCs, where each level defines a SC. This is illustrated in the following example:

*Example 1* Consider some solution vectors from $\mathbb{R}^3$, i.e. each vector is a solution consisting of three objectives $(m_1, m_2, m_3)$:

$$(0, 1, 2) \quad (1, 1, 2) \quad (2, 1, 1) \quad (7, 0, 9) \quad (8, 7, 1) \quad (1, 9, 6)$$

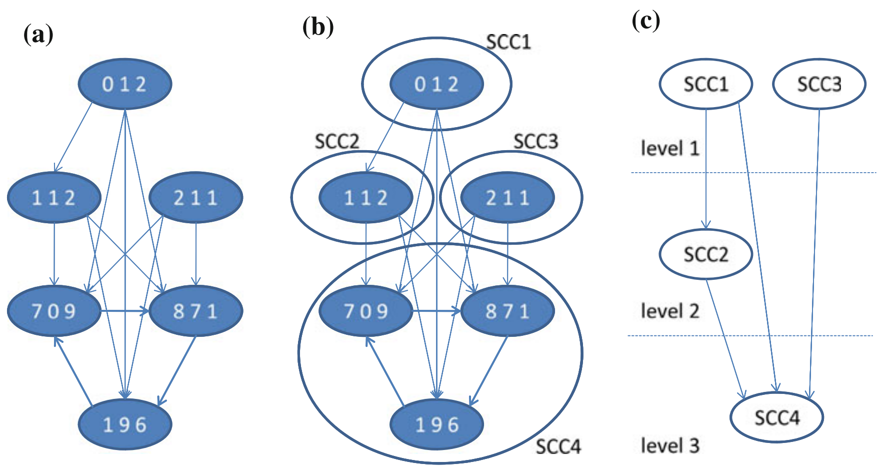

**Fig. 1** Relation graph and *Satisfiability Classes*

The relation graph of these elements and relation *Preferred* is given in Fig. 1a. Elements (0, 1, 2), (1, 1, 2) and (2, 1, 1) are preferred to the remaining elements, but (0, 1, 2) and (2, 1, 1) ((1, 1, 2) and (2, 1, 1)) are not comparable. Additionally, element (0, 1, 2) is preferred to (1, 1, 2). The remaining three vectors (8, 7, 1), (1, 9, 6), and (7, 0, 9) are pairwise comparable. But as can be seen in the relation graph they describe a "cycle". Thus relation *Preferred* is not transitive. For more details see [5].

## 2.3 Relation $\varepsilon$-Preferred

In [17] an enlargement for many-objective optimization of relation *Preferred* has been introduced. For the proposed relation $\varepsilon$-*Preferred* fitness limits $\varepsilon_i$, $1 \le i \le m$, for each dimension are defined.

**Definition 3** Let $A, B \in \boldsymbol{\Omega}$ and $\varepsilon_i$, $1 \leqslant i \leqslant m$.

$$A \prec_{\varepsilon-exceed} B \Leftrightarrow |\{i : F_i(A) < F_i(B) \wedge |F_i(A) - F_i(B)| > \varepsilon_i\}| \quad (4)$$
$$> |\{j : F_j(A) > F_j(B) \wedge |F_j(A) - F_j(B)| > \varepsilon_j\}| \quad (5)$$

$\varepsilon$-*exceed* counts how often a solution exceeds the given limits $\varepsilon_i$. Then solution A is better than solution B with respect to the limits $\varepsilon_i$, if A has more exceeding than B. Using $\varepsilon$-*exceed* the extension $\varepsilon$-*Preferred* is defined as follows:

**Definition 4** Given two solutions $A, B \in \boldsymbol{\Omega}$.

$$A \prec_{\varepsilon-preferred} B \Leftrightarrow A \prec_{\varepsilon-exceed} B \vee (B \not\prec_{\varepsilon-exceed} A \wedge A \prec_{preferred} B) \quad (6)$$

First it is counted how often a solution exceeds the $\varepsilon$-limits and the better solution is determined. If both solutions are in the given range *Preferred* is used for comparison.

*Example 2* Consider some solution vectors from $\mathbb{R}^3$, i.e. the results of three objective functions:

$$(7, 0, 9) \quad (8, 7, 1) \quad (1, 9, 6)$$

Additionally, let $\varepsilon_i = 5, 1 \leq i \leq 3$. $(7, 0, 9) \prec_{\varepsilon-preferred} (8, 7, 1)$, because for the second objective it holds $|0 - 7| > \varepsilon_2$, where solution (7, 0, 9) "wins", and for the third it holds $|9-1| > \varepsilon_3$, where solution (8, 7, 1) "wins". Since each solution has an $\varepsilon$-exceeding objective, *Preferred* is used for comparison. The same argumentation holds for $(8, 7, 1) \prec_{\varepsilon-preferred} (1, 9, 6)$ and $(1, 9, 6) \prec_{\varepsilon-preferred} (7, 0, 9)$.

Analogously to *Preferred* relation $\varepsilon$-*Preferred* is not transitive. Thus, the algorithm that computes the SCCs is applied to the relation graph as described in Sect. 2.2 and in [5].

## 2.4 Relation Prio-ε-Preferred

In many real world applications the optimization criteria have user specific preferences that have to be modeled during the optimization process. To model priorities of optimization objectives in [12] relation *Prio-ε-Preferred* is defined. It is a combination of relation $\varepsilon$-*Preferred* and a lexicographic ordering of the objectives.

Let us assume that priorities $1, 2, \ldots, k$ are assigned to the objectives in an ascending ordering, i.e. the lower the index $i$, $1 \leq i \leq k$, the higher the priority.

**Definition 5** Let $p = (p_1, \ldots, p_k)$ be a priority vector. $p_i$ determines the number of objectives that have priority $i$. The priority of an objective is calculated by the function:

$$pr : \{1, \ldots, m\} \rightarrow \{1, \ldots, k\} \tag{7}$$

The subvector of objectives $c|_i$ of priority $i$ is defined as

$$c|_i \in \mathbb{R}^{p_i}, c|_i = (c_r, \ldots, c_s) \tag{8}$$

where

$$r = \sum_{j=1}^{i-1} p_j + 1 \wedge s = \sum_{j=1}^{i} p_j. \tag{9}$$

**Fig. 2** Sketch of basic algorithm

```
Prio-ε-Preferred (population) {
 for_all (individuals)
 {
   calculate_pairwise_Prio-ε-Preferred () ;
   construct_relation_graph () ;
   calculate_strongly_connected_components () ;
   perform_level_sorting () ;
 }
 return level_of_individuals ;                //ranking
}
```

For $A, B \in \boldsymbol{\Omega}$ the relation $\prec_{\varepsilon-priopref}$ (*Prio-ε-Preferred*) is defined by

$$A \prec_{\varepsilon-priopref} B :\Leftrightarrow \exists j \in \{1, \ldots, k\} : A|_j \prec_{\varepsilon-preferred} B|_j \tag{10}$$

$$\wedge\ (\forall h < j : A|_h \not\prec_{\varepsilon-preferred} B|_h \wedge B|_h \not\prec_{\varepsilon-preferred} A|_h) \tag{11}$$

To perform a ranking of a set of elements, analogously to Sect. 2.2 the *Satisfiability Classes* are computed. For this a set of elements is pairwise compared using *Prio-ε-Preferred* and the relation graph is constructed. Then the algorithm for finding the *Strongly-Connected Components* (SCC) is applied to eliminate cycles in the relation graph. A sketch of the algorithm is given in Fig. 2.

To give an impression on the properties of relation *Prio-ε-Preferred* an example is considered.

*Example 3* Let us consider a problem with 5 objectives with 3 different priorities. Let $c = (c_1, c_2, c_3, c_4, c_5)$ a solution vector and $p = (1, 3, 1)$ a priority vector, i.e. one objective has priority 1 (i.e. $p_1 = 1$), three objectives have priority 2 ($p_2 = 3$) and one objective has priority 3 ($p_3 = 1$). This leads to the function $pr$ with $pr(1) = 1$, $pr(2) = 2$, $pr(3) = 2$, $pr(4) = 2$, and $pr(5) = 3$ what means that the first objective has priority 1, the second objective priority 2, and so on. For priority 2 the projection is $c|_2 \in \mathbb{R}^3$, $c|_2 = (c_2, c_3, c_4)$, since $r = 1 + 1 = 2$ and $s = 1 + 3 = 4$.

Now, let us consider two solution vectors, $A = (2, 7, 0, 9, 15)$ and $B = (2, 1, 9, 6, 5)$. Then it holds, that $B \prec_{\varepsilon-priopref} A$. For this, first the objectives with priority 1 are compared. Since they are equal, next the objectives with priority 2 are compared with relation $\prec_{\varepsilon-preferred}$, i.e. $(1, 9, 6) \prec_{\varepsilon-preferred} (7, 0, 9)$ (see Example 2) which leads to the statement $B \prec_{\varepsilon-priopref} A$. The last objective has not to be considered anymore, because it has lowest priority and the decision, which solution is better with respect to relation $\prec_{\varepsilon-priopref}$, has already been made.

## 3 Nurse Rostering Problem

In this section a description of the *Nurse Rostering Problem* (NRP) and the algorithm for evolutionary many-objective optimization is given.

## *3.1 Problem Description*

Since several years the NRP is of high interest and many approaches for optimization have been presented [20, 22]. The NRP is a utilization planning problem, where a working plan for employees in a hospital has to be determined. Working shifts, like e.g. day shift, night shift, stand-by shift, long shift or vacations have to be assigned to each employee and working day, such that sufficient employees are on duty and the working contracts of the employees are fulfilled.

For optimization solution schedules have to be evaluated by a fitness function. The fitness function consists of multiple criteria, that can be categorized in hard constraints that have to be fulfilled and soft constraints that improve the fitness function. A solution can still be valid, even though a soft constraint is not fulfilled. But indeed, a solution that does not fulfill soft constraints can be rejected by the planner. The constraints are given as rules that are specified in the benchmarks [20]. The benchmarks are available from [23]. The rules can be categorized into the following main areas:

1. Rules of the nurse station, e.g. sufficient nurses per shift
2. Restrictions by law, e.g. maximal hours of work per day or maximal working days per month
3. Rules resulting from ergonomics, e.g. having ergonomic shift pattern

Following the benchmarks from [20] the rule of the first category is modeled as hard constraint, whereas the rules of item 2. and 3. are given as soft constraints. The influence of the rules in the fitness function is controlled by weights that are given in the benchmarks. Concerning the weights in the benchmarks it is assumed to use a *Weighted Sum* for the calculation of the fitness function. In this application schedules for up to 16 employees for a planning period of 30 days are considered. Instead of using a weighted sum, the weights of the rules determine a priority that is used by relation *Prio-$\varepsilon$-Preferred*. The details are described in Sect. 3.2.

*Example 4* In Fig. 3 an example of a schedule for the NRP is given. To each employee A–H and day (02nd–29th) a shift is assigned, where the shifts are labeled as follows: Day shift (D), night shift (N). In the vertical columns the rules of the nurse station are evaluated. In this example for shift D three employees and for shift N one employee have to be on duty. This hard constraint is fulfilled for each day. For the maximal working days per month the vertical rows have to be evaluated. Notice, that employees E-H have half time positions, thus they have less working days. It can be seen that the ergonomic rules of the shift pattern (soft constraints) are fulfilled, i.e. desired patterns like e.g. DDDNN are given in the solution. Undesired patterns like single shifts are not scheduled. For more details about the NRP see [20, 22, 23].

| | 1 | | | | | | | 2 | | | | | | | 3 | | | | | | | 4 | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| | M | T | W | T | F | S | S | M | T | W | T | F | S | S | M | T | W | T | F | S | S | M | T | W | T | F | S | S |
| A | D | D | D | D | | | | D | D | D | D | D | | | | | D | D | D | D | D | D | | | | N | N | N |
| B | | | | D | D | D | D | N | N | | | | D | D | D | D | N | N | | | | | D | D | D | D | D | D |
| C | D | D | N | N | | | | D | D | D | D | D | | | | | | D | D | D | D | N | N | | | D | D | D |
| D | N | N | | | D | D | D | D | D | D | | | D | D | D | D | D | D | | | | D | D | N | N | | | |
| E | D | D | D | | | | | | | | | | D | D | | | | | N | N | N | | | D | D | | | |
| F | | | | | N | N | N | | | | D | D | | | D | D | D | | | | | D | D | | | | | |
| G | | | D | D | | | | | | | | N | N | N | | | | | D | D | D | | | D | D | | | |
| H | | | | | D | D | D | | | N | N | | | | N | N | | | | | | | | | | D | D | D |

**Fig. 3** Example nurse rostering schedule for benchmark GPost

## *3.2 Proposed Model for Many-Objective Optimization*

In this application an *Evolutionary Algorithm* (EA) is used to optimize the schedules. Details of representation of the individuals and evolutionary operators are left out due to page limitation.

The fitness function $F(A)$, $A \in \boldsymbol{\Omega}$, consists of $m$ functions $F_i$, $1 \le i \le m$, that are directly derived from the benchmark under consideration. For each hard and soft constraint in the benchmark an objective $F_i$ is defined that has to be minimized. Following Definition 5 the priority function $pr$ for the objectives is calculated such that the higher the weight of an objective $i$ the higher is the priority $pr(i)$. First, the objectives that correspond to hard constraints get the highest priority value 1. The remaining objectives that correspond to soft constraints get priority values depending on its weights. Objectives that have the highest weight value given in the benchmark lead to priority 2, i.e. $pr(i) = 2, \forall i$ with maximum weight. Then, the objectives with maximum weight are disregarded and again the objectives with maximum weight are considered, they get priority 3. Following this, the next weights are considered one after another. This is repeated, until a priority is assigned to each objective. The number of priorities of the considered benchmarks ranges from 2 to 10.

For the ranking of the solutions relation *Prio-ε-Preferred* is used. The solutions are compared using relation *Prio-ε-Preferred*. Then, the relation graph is constructed and the SCCs of the directed graph are computed as described in Sect. 2. For the determination of the epsilon values several methods are examined that adapt the epsilon values automatically. In Sect. 4 several methods for the justification of the epsilon values are proposed.

## 4 Approaches for $\varepsilon$-Adaptation

The justification of the epsilon values for relation *Prio-ε-Preferred* is an interesting task. For this in [12] the influence of the epsilon values in relation *Prio-ε-Preferred* has been investigated. It has been shown that the choice of the epsilon values influences the quality of the optimization. A method called AEP (*Adapted Epsilon Preferred*) has been proposed that adapts the epsilon values automatically. AEP is a straight forward method that computes the same epsilon value for all objectives. In this section more sophisticated methods for the adaptation of the epsilon values are presented. The methods examined in the experiments are described in the following:

**Adapted Epsilon Preferred (AEP)** [12]. In method AEP for all objectives one epsilon value is determined. Therefore, one individual out of the best *Satisfiability class* (SC) derived by relation *Prio-ε-Preferred* is randomly chosen. The new epsilon value is determined by the average value of all objectives of that individual:

$$\varepsilon = \sum_{j=1}^{m} \frac{Ind_{best,j}}{m} \tag{12}$$

where $Ind_{best,j}$ is the $j$th objective of an randomly chosen individual out of the best *SC*. The epsilon value is updated in each generation. The idea behind this method is that individuals can be distinguished by relation *Prio-ε-Preferred*, if the difference of the individuals exceeds the calculated average range.

**Median Epsilon Preferred (MEP)**. In method MEP one separated epsilon value for each objective is calculated. For this all individuals in a population are considered and for each objective the epsilon value is set to the median of each objective:

$$\varepsilon_j = median(\{Ind_{i,j} | 1 \leq i \leq |P|\}), 1 \leq j \leq m \tag{13}$$

where $m$ is the number of objectives, $|P|$ is the size of the population and $Ind_{i,j}$ is the $j$th objective of the $i$th individual in population $P$.

**Self-adaptation 1 (SA1)**. For each objective a separated epsilon value is calculated. First, the epsilon values are initialized using method AEP. Then in each generation a randomly chosen epsilon value is decremented. If this reduces the number of SCs, this step is revised. The idea is that a higher number of SCs leads to a meaningful ranking of the solutions.

**Self-adaptation 2 (SA2)**. Again, for each objective a separated epsilon value is calculated. The epsilon values are initialized using method AEP. Then in each generation for a randomly chosen objective the epsilon is bisected, if the set of best elements has not changed for 100 generations. The idea is to give more restriction in the ranking mechanism, if the optimization is in progress.

**Table 1** Properties of benchmarks for the NRP

| Benchmark | Rules/objectives | Priorities | Employees | Days |
|---|---|---|---|---|
| Millar-2Shift-DATA1 | 11 | 2 | 8 | 14 |
| WHPP | 12 | 2 | 30 | 14 |
| Valouxis-1 | 15 | 4 | 16 | 28 |
| GPost | 42 | 6 | 8 | 28 |
| ORTEC01 | 92 | 10 | 16 | 31 |

**Table 2** Comparison of standard methods

| | | | MOO model [12] | Proposed MOO model | |
|---|---|---|---|---|---|
| Benchmark | Objectives | Weighted Sum | AEP | NSGA-II | AEP |
| Millar-2Shift-DATA1 | 11 | 1310 | 1590 | 1390 | 1190 |
| WHPP | 12 | 29 | - | 133 | 27 |
| Valouxis-1 | 15 | 13986 | 16692 | 148500 | 13542 |
| GPost | 42 | 7159 | 7557 | 26528 | 11830 |
| ORTEC01 | 92 | 9132 | 11672 | 36740 | 10040 |

## 5 Experimental Results

In this section the experimental results of the presented approaches are described. The benchmarks for the NRP are taken from [23]. and its properties are summarized in Table 1. The optimization rules given in the benchmark directly correspond to the objectives (column *Rules/Objectives*). The number of different priorities of the objectives are given in column *Priorities*. Columns *Employees* and *Days* show the benchmarks' number of employees and the planning period, respectively. For each benchmark and for each method presented in Sect. 4 the EA is run 10 times with different random seeds. Then, the average value over these 10 runs is calculated. The population size is set to 50 and the EA runs for 5000 generations. The average values of the presented approaches are given in Tables 2 and 3. The methods are compared using the *Weighted Sum*. For this, the objectives are transferred into a single objective fitness function, such that the weights in the benchmarks are taken to weight each objective.[2]

In a first series of experiments the proposed model for MOO, where for each rule an objective is defined, is compared to the restricted model from [12]. There only hard and soft constraints are considered as optimization objectives. For both models method AEP from [12] is applied to the NRP. The average values can be seen in columns *AEP* of Table 2. The results can be improved, if a refinement of the model for MOO as proposed in this paper is performed.

[2]Originally the benchmarks are designed for optimization using a *Weighted Sum*. Thus, the weights are justified by a planner and directly given in the benchmark.

**Table 3** Comparison of $\varepsilon$-adaptation methods for the proposed MOO model

| Benchmark | Objectives | Weighted Sum | NSGA-II | AEP | MEP | SA1 | SA2 |
|---|---|---|---|---|---|---|---|
| Millar-2Shift-DATA-1 | 11 | 1310 | 1390 | 1190 | 1540 | 1280 | 1220 |
| WHPP | 12 | 29 | 133 | 27 | 32 | 27 | 24 |
| Valouxis-1 | 15 | 13986 | 148500 | 13542 | 12890 | 11852 | 13132 |
| GPost | 42 | 7159 | 26528 | 11830 | 12300 | 13006 | 11401 |
| ORTEC01 | 92 | 9132 | 36740 | 10040 | 10927 | 11549 | 10618 |

Additionally, AEP is compared to NSGA-II [3], which is a basic method in evolutionary multi-objective optimization (see column *NSGA-II*). For comparison, analogously to *Prio-$\varepsilon$-Preferred* NSGA-II is extended such that it can also handle priorities.[3] Thus, it is comparable to the methods that are based on relation *Prio-$\varepsilon$-Preferred*. Furthermore, it can be seen that AEP outperforms NSGA-II for each considered benchmark. Especially for *Valouxis-1* an improvement of more than 90 % can be observed. Furthermore, a comparison to an approach that is based on *Weighted Sums* is given. It is a single objective evolutionary algorithm, where the weights in the benchmarks are used to calculate the fitness function. The comparison shows that for most benchmarks the overall quality has been improved.

In a next series of experiments the approaches for adaptation of the epsilon values presented in this paper are compared. The results are summarized in columns *MEP*, *SA1* and *SA2* of Table 3. A comparison to NSGA-II in Table 2 shows that MEP fails only for one example (*Millar-2Shift-DATA1*), whereas the self-adaptive approaches SA1 and SA2 improve NSGA-II for all benchmarks. For three out of the considered benchmarks both methods compute better results than the weighted sum approach. Notice, the benchmarks are designed such that optimization with *Weighted Sums* can easily be performed, i.e. the weights are specified in the benchmark. Thus, even these results can be improved, if the full potential of the proposed model for many-objective optimization is used. Only for benchmarks *GPost* and *ORTEC01* SA1 and SA2 fail to calculate the best results. Both benchmarks consist of 42 and more objectives. A comparison shows that AEP can be improved using the self-adapting techniques SA1 and SA2. It is focus of current work to investigate the self-adapting techniques such that also problems with a higher number of objectives are solved sufficiently.

To give an impression on the quality of the priority based optimization presented above a solution element out of the best SC derived by SA1 is compared to an element from the non-dominated set derived by NSGA-II. In Figs. 4 and 5 a comparison for benchmarks *Valouxis-1* and *WHPP* is shown. The objectives and its priorities are

[3]The main reason for using NSGA-II for comparison is that it can easily be enlarged such that it can handle priorities as described in this paper. Other methods like e.g. Hype [4] are more suitable for *Many-Objective Optimization*, but it is not obvious how to incorporate the priorities. This is an interesting task for further developments.

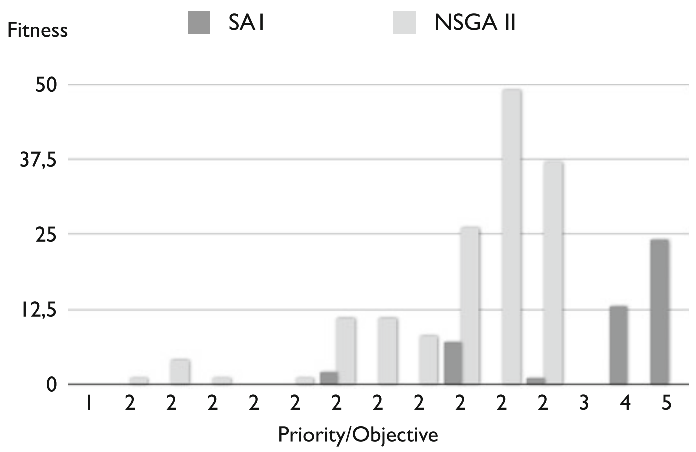

**Fig. 4** Comparison of solutions: benchmark Valouxis-1

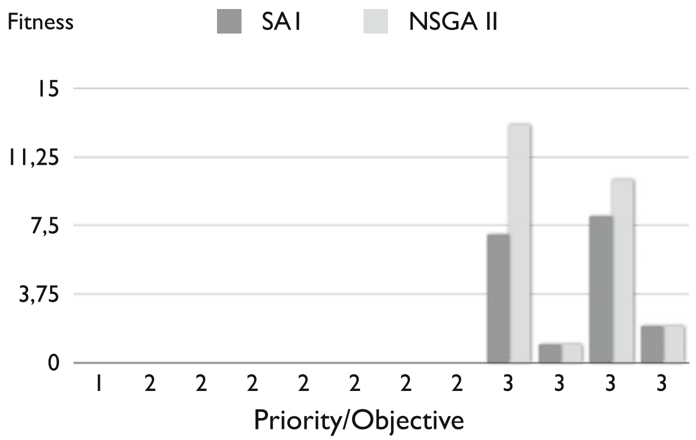

**Fig. 5** Comparison of solutions: benchmark WHPP

specified at the x-axis. For benchmark *Valouxis-1* it can be seen, that the solution obtained by SA1 has only 5 objectives over zero, whereas NSGA-II has 10 objectives over zero. This means that the solution obtained by SA1 is *Preferred* to the solution obtained by method NSGA-II. Additionally, if the absolute values of the objectives are compared, SA1 performs better than NSGA-II. The same observation holds for benchmark *WHPP*. For this example the objectives with priorities 1 and 2 are solved optimally. For objectives with priority 3 it can be observed that the solution obtained by SA1 even *Dominates* the solution from NSGA-II.

## 6 Conclusions

In this paper a model for *Many-Objective Optimization*, i.e. optimization problems with more than three objectives, based on the *Prio-ε-Preferred* relation has been investigated. For this, heuristics for the determination of the epsilon values are presented. The model is applied to the *Nurse Rostering Problem*, a resource planning problem where different working shifts have to be assigned to the nurses in a hospital. To compare the proposed methods experiments on benchmark examples are performed. It turned out that using self-adapting mechanisms for the adaption of the epsilon value NSGA-II and an approach based on *Weighted Sums* can be outperformed.

**Acknowledgments** I'd like to thank André Sülflow and Rolf Drechsler for helpful discussions and comments and for their contributions to previous work.

## References

1. Fonseca, C., Fleming, P.: An overview of evolutionary algorithms in multiobjective optimization. Evol. Comput. **3**(1), 1–16 (1995)
2. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans. Evol. Comput. **3**(4), 257–271 (1999)
3. Deb, K.: Multi-objective Optimization using Evolutionary Algorithms. Wiley, New York (2001)
4. Bader, J., Zitzler, E.: HypE: an algorithm for fast hypervolume-based many-objective optimization. Evol. Comput. **19**(1), 45–76 (2011)
5. Drechsler, N., Drechsler, R., Becker, B.: Multi-objective optimisation based on relation favour. In: International Conference on Evolutionary Multi-Criterion Optimization, pp. 154–166 (2001)
6. Hughes, E.: Radar waveform optimization as a many-objective application benchmark. In: International Conference on Evolutionary Multi-Criterion Optimization, pp. 700–714 (2007)
7. Pizzuti, C.: A multiobjective genetic algorithm to find communities in complex networks. IEEE Trans. Evol. Comput. **16**(3), 418–430 (2012)
8. Schmiedle, F., Drechsler, N., Große, D., Drechsler, R.: Priorities in multi-objective optimization for genetic programming. In: Genetic and Evolutionary Computation Conference, pp. 129–136 (2001)
9. Wickramasinghe, U., Li, X.: A distance metric for evolutionary many-objective optimization algorithms using user-preferences. In: 22nd Australasian Joint Conference on Advances in Artificial Intelligence (AI'09), pp. 443–453 (2009)
10. Auger, A., Bader, J., Brockhoff, D., Zitzler, E.: Articulating user preferences in many-objective problems by sampling the weighted hypervolume. In: Genetic and Evolutionary Computation Conference, pp. 555–562 (2009)
11. Wagner, T., Trautmann, H.: Integration of preferences in hypervolume-based multiobjective evolutionary algorithms by means of desirability functions. IEEE Trans. Evol. Comput. **14**(5), 688–701 (2012)
12. Drechsler, N., Sülflow, S., Drechsler, R.: Incorporating user preferences in many-objective optimization using relation ε-preferred. In: International Conference on Evolutionary Computation Theory and Applications (2013)
13. Geiger, M.: Multi-criteria curriculum-based course timetabling—a comparison of a weighted sum and a reference point based approach. In: International Conference on Evolutionary Multi-Criterion Optimization, pp. 290–304 (2009)

14. Goldberg, D.: Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley Publisher Company, Inc, Reading (1989)
15. Corne, D., Knowles, J.: Techniques for highly multiobjective optimization: theorie and applications. In: Genetic and Evolutionary Computation Conference, pp. 773–780 (2007)
16. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Evolutionary many-objective optimization: a short review. In: IEEE Congress on Evolutionary Computation, pp. 2424–2431 (2008)
17. Sülflow, A., Drechsler, N., Drechsler, R.: Robust multi-objective optimization in high-dimensional spaces. In: International Conference on Evolutionary Multi-Criterion Optimization, pp. 715–726 (2007)
18. Zhang, Q., Li, H.: Moea/d: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
19. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multi-objective optimization. Technical Report 112, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland (2001)
20. Burke, E., Curtois, T., Qu, R., Vanden-Berghe, G.: Problem model for nurse rostering benchmark instances. Technical report, ASAP, School of Computer Science, University of Nottingham, UK (2012)
21. Cormen, T., Leierson, C., Rivest, R.: Introduction to Algorithms. MIT Press, Cambridge (1990)
22. Burke, E., Causmaecker, P.D., Berghe, G., Landeghem, H.: The state of the art of nurse rostering. J. Sched. **7**, 441–499 (2004)
23. Benchmarks: Employee scheduling benchmark data set: Technical report, ASAP, School of Computer Science, The University of Nottingham, UK (2012). http://www.cs.nott.ac.uk/~tec/nrp/