GDE3: The third Evolution Step of Generalized Differential Evolution

Saku Kukkonen and **Jouni Lampinen**
Department of Information Technology
Lappeenranta University of Technology
P.O. Box 20, FIN-53851 Lappeenranta, Finland
saku.kukkonen@lut.fi

Abstract-

A developed version of Generalized Differential Evolution, GDE3, is proposed. GDE3 is an extension of Differential Evolution (DE) for global optimization with an arbitrary number of objectives and constraints. In the case of a problem with a single objective and without constraints GDE3 falls back to the original DE. GDE3 improves earlier GDE versions in the case of multi-objective problems by giving a better distributed solution. Performance of GDE3 is demonstrated with a set of test problems and the results are compared with other methods.

1 Introduction

During the last 15 years, Evolutionary Algorithms (EAs) have gained popularity in solving difficult multi-objective optimization problems (MOOPs) since EAs are capable of dealing with objective functions, which are not mathematically well behaving, *e.g.*, discontinuous, non-convex, multi-modal, and non-differentiable. Multi-objective EAs (MOEAs) are also capable of providing multiple solution candidates in a single run, which is desirable with MOOPs.

Differential Evolution (DE) is a relatively new EA and it has been gaining popularity during previous years. Several extensions of DE for multi-objective optimization have already been proposed. Some basic approaches just convert MOOPs to single-objective forms and use DE to solve these [3, 5, 36].

The first method extending DE for multi-objective optimization using the Pareto approach was Pareto-based DE approach [6]. Pareto Differential Evolution [4] was also mentioned about the same time, unfortunately without an explicit description of the method. After these, the Pareto(-frontier) Differential Evolution (PDE) algorithm [2] and a first version of Generalized Differential Evolution (GDE) [22] were introduced. Later on, Self-adaptive PDE (SPDE) [1], the Pareto DE Approach (PDEA) [26], Adaptive Pareto DE (APDE) [37], Multi-Objective DE (MODE) [13], and Vector Evaluated DE (VEDE) [29] have been proposed. The latest proposals are a second version of GDE [21] and DE for Multiobjective Optimization (DEMO) [32]. Research demonstrating the performance of PDEA over the elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II) [9] with rotated MOOPs has also been reported [17].

Besides solving problems with multiple objectives, DE has also been modified for handling problems with constraints [7, 23, 25, 27, 33, 35]. Most of these are based on applying penalty functions.

Earlier GDE versions had already the ability to handle any number of objectives and constraints. The latest version, GDE3, introduced in this paper is an attempt to improve earlier versions in the case of multiple objectives.

2 Multi-Objective Optimization with Constraints

Many practical problems have multiple objectives and several aspects cause multiple constraints to problems. For example, mechanical design problems have several objectives such as obtained performance and manufacturing costs, and available resources may cause limitations. Constraints can be divided into boundary constraints and constraint functions. Boundary constraints are used when the value of a decision variable is limited to some range, and constraint functions represent more complicated constraints, which are expressed as functions.

A mathematically constrained MOOP can be presented in the form:

$$\begin{array}{ll} \text{minimize} & \{f_1(\vec{x}), f_2(\vec{x}), \dots, f_M(\vec{x})\} \\ \text{subject to} & (g_1(\vec{x}), g_2(\vec{x}), \dots, g_K(\vec{x}))^T \leq \vec{0}. \end{array} \quad (1)$$

Thus, there are M functions to be optimized and K constraint functions. Maximization problems can be easily transformed to minimization problems and different constraints can be converted to form $g_j(\vec{x}) \leq 0$, Thereby the formulation in Eq. 1 is without loss of generality.

Typically, MOOPs are often converted to single-objective optimization problems by predefining weighting factors for different objectives, expressing the relative importance of each objective. Optimizing several objectives simultaneously without articulating the relative importance of each objective *a priori*, is often called Pareto-optimization. An obtained solution is Pareto-optimal if none of the objectives can be improved without impairing at least one other objective [28, p. 11]. If the obtained solution can be improved in such a way that at least one objective improves and the other objectives do not decline, then the new solution dominates the original solution. The objective of Pareto-optimization is to find a set of solutions that are not dominated by any other solution.

A set of Pareto-optimal solutions form a Pareto front, and an approximation of the Pareto front is called a set of non-dominated solutions. From the set of non-dominated solutions the decision-maker may pick a solution, which provides a suitable compromise between the objectives. This can be viewed as *a posteriori* articulation of the decision-makers preferences concerning the relative importance of each objective.

Later on in this paper, the obtained non-dominated set is referred to as a *solution*, and a member of a non-dominated set or a population is referred to as a *vector* to distinguish these.

Weak dominance relation $\preceq$ between two vectors is defined such that $\vec{x}_1$ weakly dominates $\vec{x}_2$, *i.e.*, $\vec{x}_1 \preceq \vec{x}_2$ iff $\forall i : f_i(\vec{x}_1) \leq f_i(\vec{x}_2)$. Dominance relation $\prec$ between two vectors is defined such that $\vec{x}_1$ dominates $\vec{x}_2$, *i.e.*, $\vec{x}_1 \prec \vec{x}_2$ iff $\vec{x}_1 \preceq \vec{x}_2 \ \wedge \ \exists i : f_i(\vec{x}_1) < f_i(\vec{x}_2)$. The dominance relationship can be extended to take into consideration constraint values besides objective values. A constraint-domination $\prec_c$ is defined here so that $\vec{x}_1$ constraint-dominates $\vec{x}_2$, *i.e.*, $\vec{x}_1 \prec_c \vec{x}_2$ iff any of the following conditions is true [22]:

- $\vec{x}_1$ is feasible and $\vec{x}_2$ is not.
- $\vec{x}_1$ and $\vec{x}_2$ are infeasible and $\vec{x}_1$ dominates $\vec{x}_2$ in constraint function space.
- $\vec{x}_1$ and $\vec{x}_2$ are feasible and $\vec{x}_1$ dominates $\vec{x}_2$ in objective function space.

The definition for weak constraint-domination $\preceq_c$ is analogous. This constraint-domination definition differs from the approach presented in [8, pp. 301–302] in the case of two infeasible vectors and was developed independently.

3 Differential Evolution

The DE algorithm [31, 34] was introduced by Storn and Price in 1995. Design principles in DE were simplicity, efficiency, and the use of floating-point encoding instead of binary numbers. Like a typical EA, DE has some random initial population, which is then improved using selection, mutation, and crossover operations. Several methods exist to determine a stopping criterion for EAs but usually a predefined upper limit for the number of generations or function evaluations to be computed provides an appropriate stopping condition.

In each generation DE goes through each decision vector $\vec{x}_{i,G}$ of the population and creates a corresponding trial vector $\vec{u}_{i,G}$. Here, i is an index of the vector in the population and G is a generation index. Creation of the trial vector is done as follows in the most common DE version, DE/rand/1/bin [30]:

```
$r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$, (randomly selected,
  except mutually different and different from $i$)
$j_{rand} = \text{floor}(rand_i[0,1) \cdot D) + 1$
for($j = 1; j \leq D; j = j + 1$)
{
    if($rand_j[0,1) < CR \vee j = j_{rand}$)
        $u_{j,i,G} = x_{j,r_3,G} + F \cdot (x_{j,r_1,G} - x_{j,r_2,G})$
    else
        $u_{j,i,G} = x_{j,i,G}$
}
                                                        (2)
```

The scaled difference between two randomly chosen vectors, $F \cdot (\vec{x}_{r_1,G} - \vec{x}_{r_2,G})$, defines magnitude and direction of the mutation. When the difference is added to a third randomly chosen vector $\vec{x}_{r_3,G}$, this corresponds to the mutation of the third vector.

Both CR and F are user defined control parameters for the DE algorithm and they remain fixed during the whole execution of the algorithm. Parameter CR, controlling the crossover operation, represents the probability that an element for the trial vector is chosen from a linear combination of three randomly chosen vectors instead of from the old vector $\vec{x}_{i,G}$. The condition "$j = j_{rand}$" is to make sure that at least one element is different compared to elements of the old vector. Parameter F is a scaling factor for mutation and its value is typically $(0, 1+]$. In practice, CR controls the rotational invariance of the search, and its small value (e.g. 0.1) is practicable with separable problems while larger values (e.g. 0.9) are for non-separable problems. Control parameter F controls the speed and robustness of the search, *i.e.*, a lower value for F increases the convergence rate but also the risk of stacking into a local optimum.

The basic idea of DE is that the mutation is self-adaptive to the objective function surface and to the current population in the same way as in Covariance Matrix Adaptation Evolutionary Strategies (CMA-ES) [16] but without the computational burden of covariance matrix calculations that are scaling unfavorably with the dimensionality of the problem. At the beginning of generations the magnitude of the mutation is large because vectors in the population are far away in the search space. When evolution proceeds and the population converges, the magnitude of the mutation gets smaller. The self-adaptability of DE permits a global search

A trial vector $\vec{u}_{i,G}$ created by mutation and crossover operations is compared to an old vector $\vec{x}_{i,G}$. If the trial vector has an equal or better objective value, then it replaces the old vector in the next generation. Therefore, the average objective value of the population will never worsen making DE an elitist method.

4 Generalized Differential Evolution

The first version of a Generalized Differential Evolution (GDE) extended DE for constrained multi-objective optimization and was obtained by modifying the selection rule of the basic DE [22]. The basic idea in the selection rule was that the trial vector was selected to replace the old vector in the next generation if it weakly constraint-dominated the old vector. There was no sorting of non-dominated vectors during the optimization process or any mechanism for maintaining the distribution and extent of the solution. Also, there was no extra repository for non-dominated vectors. Still, GDE was able to provide a surprisingly good solution but was too sensitive for the selection of the control parameters [20].

Later on GDE was modified to make a decision based on the crowdedness when the trial and old vector were feasible and non-dominating each other in the objective function space [21]. This improved the extent and distribution of the solution but slowed down the convergence of the overall population because it favored isolated vectors far from the Pareto front before all the vectors were converged near the Pareto front. This version, GDE2, was still too sensitive for the selection of the control parameters.

The third version of GDE proposed in this paper extending the DE/rand/1/bin method to problems with M objectives and K constraint functions formally presented in Eq. 3.

```
Input :$D, G_{max}, NP \geq 4, F \in (0, 1+], CR \in [0,1]$, and initial bounds: $\vec{x}^{(lo)}, \vec{x}^{(hi)}$
Initialize : $\left\{ \begin{array}{l} \forall i \leq NP \wedge \forall j \leq D : x_{j,i,G=0} = x_j^{(lo)} + rand_j[0,1] \cdot \left( x_j^{(hi)} - x_j^{(lo)} \right) \\ i = \{1,2,\ldots,NP\}, j = \{1,2,\ldots,D\}, G = 0, m = 0, rand_j[0,1) \in [0,1), \end{array} \right.$

While $G < G_{max}$
    $\forall i \leq NP$
        Mutate and recombine:
        $r_1, r_2, r_3 \in \{1,2,\ldots,NP\}$, randomly selected,
                except mutually different and different from $i$
        $j_{rand} \in \{1,2,\ldots,D\}$, randomly selected for each $i$
        $\forall j \leq D, u_{j,i,G} = \left\{ \begin{array}{l} x_{j,r_3,G} + F \cdot (x_{j,r_1,G} - x_{j,r_2,G}) \\ \qquad \text{if} \quad rand_j[0,1) < CR \vee j = j_{rand} \\ x_{j,i,G} \quad \text{otherwise} \end{array} \right.$
        Select :
        $\vec{x}_{i,G+1} = \left\{ \begin{array}{ll} \vec{u}_{i,G} & \text{if} \quad \vec{u}_{i,G} \preceq_c \vec{x}_{i,G} \\ \vec{x}_{i,G} & \text{otherwise} \end{array} \right.$
        Set :
            $m = m + 1$
            $\vec{x}_{NP+m,G+1} = \vec{u}_{i,G}$
            if $\left\{ \begin{array}{l} \forall j : g_j(\vec{u}_{i,G}) \leq 0 \\ \wedge \\ \vec{x}_{i,G+1} == \vec{x}_{i,G} \\ \wedge \\ \vec{x}_{i,G} \not\prec \vec{u}_{i,G} \end{array} \right.$
    While $m > 0$
        Select $\vec{x} \in \{\vec{x}_{1,G+1}, \vec{x}_{2,G+1}, \ldots, \vec{x}_{NP+m,G+1}\}$ :
            $\left\{ \begin{array}{l} \forall i \quad \vec{x} \not\prec_c \vec{x}_{i,G+1} \\ \wedge \\ \forall(\vec{x}_{i,G+1} : \vec{x}_{i,G+1} \not\prec_c \vec{x}) \quad CD(\vec{x}) \leq CD(\vec{x}_{i,G+1}) \end{array} \right.$
        Remove $\vec{x}$
        $m = m - 1$
    $G = G + 1$
                                                                                  (3)
```

Notation *CD* means Crowding Distance [9], which approximates the crowdedness of a vector in its non-dominated set. Also, some other distance measure for crowdedness could be used. The parts that are new compared to previous GDE versions are framed in Eq. 3. Without these parts, the algorithm is identical to the first GDE version. Later on in this paper the proposed method given in Eq. 3 is called the *Generalized Differential Evolution 3 (GDE3)*. It handles any number of M objectives and any number of K constraints, including the cases where $M = 0$ (constraint satisfaction problem) and $K = 0$ (unconstrained problem), and the original DE is a special case of GDE3. GDE3 can been seen as a combination of earlier GDE versions and PDEA [26]. A similar approach was also proposed in DEMO [32] without constraint handling, and DEMO does not fall back to the original DE in the case of single objective as GDE3 does.

Selection in GDE3 is based on the following rules:

- In the case of infeasible vectors, the trial vector is selected if it weakly dominates the old vector in constraint violation space, otherwise the old vector is selected.
- In the case of the feasible and infeasible vectors, the feasible vector is selected.
- If both vectors are feasible, then the trial is selected if it weakly dominates the old vector in the objective function space. If the old vector dominates the trial vector, then the old vector is selected. If neither vector dominates each other in the objective function space, then both vectors are selected for the next generation.

After a generation, the size of the population may have been increased. If this is the case it is then decreased back to the original size based on a similar selection approach used in NSGA-II. Vectors are sorted based on non-dominance and crowdedness. The worst population members according to these measurements are removed to decrease the size of the population to the original size. Non-dominated sorting is modified to take into consideration also constraints, and selection based on Crowding Distance is improved over the original method of NSGA-II to provide a better distributed set of vectors [19].

When $M = 1$ and $K = 0$, there are no constraints to be evaluated and the selection is simply

$$\vec{x}_{i,G+1} = \begin{cases} \vec{u}_{i,G} & \text{if} \quad f(\vec{u}_{i,G}) \leq f(\vec{x}_{i,G}) \\ \vec{x}_{i,G} & \text{otherwise} \end{cases} , \qquad (4)$$

which is same as for the original DE. The population is not increased because this requires that $\vec{u}_{i,G}$ and $\vec{x}_{i,G}$ do not weakly dominate each other, which cannot be true in the case of a single objective. Since the population is not increased, there is no need to remove elements. Therefore, GDE3 is identical to the original DE in this case. This makes it possible to change DE/rand/1/bin strategy to any other DE strategy such as presented in [14, 37] or, generally, to any method where a child vector is compared against a parent vector and a better one is preserved.

In NSGA-II and PDEA, the size of the population after a generation is $2NP$, which is then decreased to size NP. In GDE3 and DEMO, the size of the population after a generation is between NP and $2NP$ because the size of the population is increased only if the trial and the old vector are feasible and do not dominate each other.[1] Decreasing the size of the population at the end of a generation is the most complex operation in the algorithm. This needs non-dominated sorting, which in GDE3 uses the concept of constraint-domination defined in 2. Non-dominated sorting can be implemented to run in time $O\left(N \log^{M-1} N\right)$ [18]. Also, niching is done for non-dominated members of the population, which is a complex operation if clustering techniques are applied. Instead of clustering, niching is performed using an approximate distance measure, Crowding Distance, which can be calculated in time $O\left(MN \log N\right)$ [18]. Overall running time for GDE3 in Eq. 3 is $O\left(G_{max} N \log^{M-1} N\right)$ for large N.

GDE3 can be implemented in such a way that the number of function evaluations is reduced because not always all the constraints and objectives need to be evaluated, *e.g.*, inspecting constraint violations (even one constraint) is often enough to determine, which vector to select for the next generation [23, 31]. However, in the case of feasible vectors all the objectives need to be evaluated.

5 Experiments

GDE3 was evaluated with a set of test problems available from the literature [8, 10, 11]. The idea was to select known representative problems from different problem type categories. In repeated tests, a standard two-sample t-test was used to evaluate the significance of the obtained numerical results. Suitable control parameter values of GDE3 for each problem were found based on problem characteristics and by trying out a couple of different control parameter values.

[1]GDE3 could be modifi ed to preserve the old and the trial vector in the case of constrained-non-domination, but this would increase number of function evaluations needed and slow down convergence.

5.1 Singe-Objective Optimization

The performance of GDE3 in the case of single-objective optimization is illustrated with two classical multi-modal test problems, Rastrigin's and Schwefel's functions with 20 variables. Since both problems are separable, a low value for CR was used. The control parameter value F was set as low as possible while still obtaining a global optimum. The control parameters were $CR = 0.0$, $F = 0.5$ for Rastrigin's function and $CR = 0.2$, $F = 0.4$ for Schwefel's function.

The test functions, initialization ranges, used population sizes, desired target values, and a number of needed function evaluations as shown in Table 1. A minimum, mean, and maximum number of function evaluations after 100 independent runs are reported. The number of needed function evaluations were significantly smaller than with the Omni-Optimizer reported in [12].

5.2 Bi-Objective Test Problem

Improved selection based on the Crowding Distance is demonstrated with a simple bi-objective optimization problem, which is defined as [8, p. 176]:

$$\begin{array}{ll} \text{Minimize} & f_1(\vec{x}) = x_1 \\ \text{Minimize} & f_2(\vec{x}) = \frac{1+x_2}{x_1} \\ \text{subject to} & x_1 \in [0.1, 1], x_2 \in [0, 5] \end{array} \qquad (5)$$

This problem is relatively easy to solve for MOEAs, and GDE3 finds a solution converged to the Pareto front in about 20 generations. The problem was solved with GDE3 and NSGA-II having a population size of 100 and 500 generations. Control parameters for GDE3 were $CR = 0.2$ and $F = 0.2$, and for NSGA-II $p_c = 0.9$, $p_m = 1/D$, $\eta_c = 20$, and $\eta_m = 20$ [9]. A large number of generations were used to make sure that the obtained solution converged to the Pareto front and only the diversity of the solution was measured. Results after one run are shown in Figure 1. A better distribution obtained for the solution for GDE3 than NSGA-II can be observed with a careful view.

The problem was solved 100 times and diversity was measured using spacing (S) [8, pp. 327–328], which measures the standard deviation of the distances from each vector to the nearest vector in the obtained non-dominated set. A small value for S is better, and $S = 0$ for ideal distribution. Mean and standard deviations for spacing are 0.0030 ± 0.0003 and 0.0074 ± 0.0007 for GDE3 and NSGA-II, respectively. GDE3 has more than double the lower spacing value than NSGA-II has, *i.e.*, GDE3 obtains a better distributed solution than NSGA-II in the case of this problem.

5.3 Bi-Objective Mechanical Design Problem

A bi-objective spring design problem [8, pp. 453–455] was selected to demonstrate the GDE3's ability to handle several

Function	$f(\vec{x})$	D	Range	N	f^*	Min	Mean	Max
Rastrigin	$\sum_{i=1}^{D} x_i{}^2 + 10\left(1-\cos(2\pi x_i)\right)$	20	[-10, 10]	20	0.01	8029 (19260)	9085 (24660)	10184 (29120)
Schwefel	$418.982887D - \sum_{i=1}^{D} x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	20	[-500, 500]	50	0.01	14996 (54950)	16540 (69650)	18479 (103350)

Table 1: Single-objective optimization problems, initialization ranges, population size, desired target value, and the needed number of function evaluations for GDE3. Results reported in [12] for the Omni-Optimizer are in parenthesis.

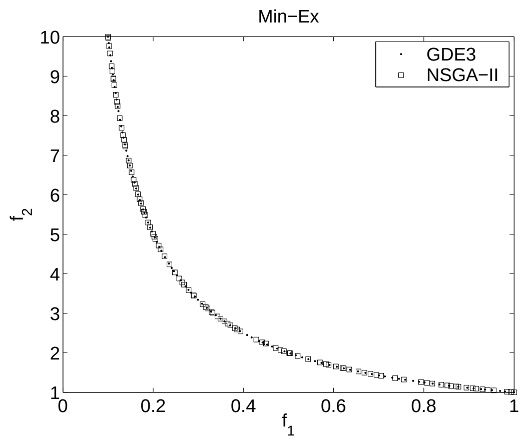

Figure 1: A simple bi-objective optimization problem solved with GDE3 and NSGA-II.

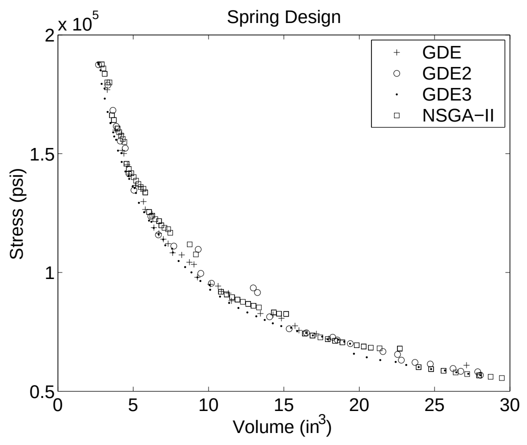

Figure 2: The spring design problem solved with GDE, GDE2, GDE3 and NSGA-II.

constraints and different types of decision variables. GDE3 uses real-coded variables as genotypes, which are converted to corresponding phenotypes before evaluation of the objective and constraint functions [31].

The problem is to design a helical compression spring, which has a minimum volume and minimal stress. Objective functions are nonlinear and the problem has three variables: the number of spring coils x_1 (integer), the wire diameter x_2 (discrete having 42 non-equispaced values), and the mean coil diameter x_3 (real). Besides the boundary constraints, the problem has eight inequality constraint functions from which most are nonlinear.

Results for the different GDE versions and NSGA-II after a single run are shown in Figure 2. The size of the population and the number of generations were 100 for the methods. Control parameter values for GDEs were $CR = 0.9$ and $F = 0.5$. The control parameters for NSGA-II were $p_c = 1.0$, $p_m = 1/D$, $\eta_c = 10$, and $\eta_m = 500$ used in [8, pp. 450]. The number of needed function evaluations for the GDEs are reported in Table 2. NSGA-II needed 10000 function evaluations for each objective and constraint function.

In preliminary tests GDE3 was found to be more stable than earlier GDE versions for the selection of the control parameters. In these tests, GDE and GDE2 also performed poorer compared to GDE3 and therefore they were excluded from further comparison in this paper.

5.4 Constrained Bi-Objective Test Problems

Constrained bi-objective test problems CTP1 and CTP2 [10] having $D = 6$, $x_i \in [0, 1]$, and function $g(\vec{x}) = 1 + \sum_{j=2}^{D} x_j{}^2$ controlling difficulty to converge to the Pareto front were used. These problems were solved 100 times. The size of the population was 100 and the number of generations was 50. Control parameters for GDE3 were $CR = 0.9$ and $F = 0.1$, and for NSGA-II $p_c = 0.9$, $p_m = 1/D$, $\eta_c = 20$, and $\eta_m = 20$ used in [10].

Results were compared using spacing and binary metrics set coverage $\mathcal{C}$ metric [8, pp. 325–326] and a $\mathcal{V}$ measure [15, 24]. The $\mathcal{C}(\mathrm{A}, \mathrm{B})$ metric measures the fraction of members of B that are dominated by members of A. The $\mathcal{V}(\mathrm{A}, \mathrm{B})$ measures the fraction of the volume of the minimal hypercube containing both fronts that is dominated by the members of A but is not dominated by the members of B. Greater values for $\mathcal{C}$ and the $\mathcal{V}$ metrics are desirable.

The results shown in Table 3. With CTP1, spacing (S) shows strongly and $\mathcal{V}$ metric slightly that GDE3 performs better but $\mathcal{C}$ metric shows strongly opposite implying that NSGA-II has converged closer to the Pareto front. With CTP2, there is no significant difference between obtained S values and the binary metrics show contradicting results. Contradicting results for binary metrics are due to the fact that the $\mathcal{C}$ metric emphasizes convergence over diversity whereas the $\mathcal{V}$ metric considers both issues.

	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	f_1	f_2
GDE	10100	8990	8901	8778	8115	8115	5704	5529	5515	2920
GDE2	10100	8504	8419	8008	7741	7741	6348	5874	5846	5846
GDE3	10100	8961	8879	8548	8319	8317	4885	4587	4566	4566

Table 2: Number of needed constraint (g_j) and objective (f_i) function evaluations needed by GDE, GDE2, and GDE3 for the spring design problem.

	$S(\mathrm{G})$	$S(\mathrm{N})$	$\mathcal{C}(\mathrm{G},\mathrm{N})$	$\mathcal{C}(\mathrm{N},\mathrm{G})$	$\mathcal{V}(\mathrm{G},\mathrm{N})$	$\mathcal{V}(\mathrm{N},\mathrm{G})$
CTP1	$\mathbf{0.0048}\pm$ $\mathbf{0.0039}$	$0.0075 \pm$ 0.0026	$0.1303 \pm$ 0.0415	$\mathbf{0.2023}\pm$ $\mathbf{0.0904}$	$\mathbf{0.0034}\pm$ $\mathbf{0.0015}$	$0.0027 \pm$ 0.0016
CTP2	$\mathbf{0.0092}\pm$ $\mathbf{0.0073}$	$0.0113 \pm$ 0.0085	$0.2655 \pm$ 0.0742	$\mathbf{0.3588}\pm$ $\mathbf{0.0685}$	$\mathbf{0.0031}\pm$ $\mathbf{0.0011}$	$0.0022 \pm$ 0.0006
DTLZ1	$\mathbf{0.0179}\pm$ $\mathbf{0.0007}$	$0.0274 \pm$ 0.0171	$\mathbf{0.3842}\pm$ $\mathbf{0.1449}$	$0.0021 \pm$ 0.0100	$\mathbf{0.0046}\pm$ $\mathbf{0.0033}$	$0.0012 \pm$ 0.0011
DTLZ4	$\mathbf{0.0214}\pm$ $\mathbf{0.0010}$	$0.0238 \pm$ 0.0009	$\mathbf{0.0948}\pm$ $\mathbf{0.0240}$	$0.0123 \pm$ 0.0066	$\mathbf{0.0085}\pm$ $\mathbf{0.0007}$	$0.0059 \pm$ 0.0008

Table 3: Spacing (S), $\mathcal{C}$, and $\mathcal{V}$ metrics for the CTP and DTLZ problems (G = GDE3 and N = NSGA-II).

5.5 Tri-Objective Test Problems

Finally, GDE3 was used to solve problems with more than two objectives. Tri-objective test problems DTLZ1 and DTLZ4 [11] were selected for this purpose. The size of the population was 500 and the number of generations was 150 for DTLZ1 and 50 for DTLZ4. Control parameters for GDE3 were $CR = 0.2$ and $F = 0.2$, and for NSGA-II $p_c = 1.0$, $p_m = 1/D$, $\eta_c = 15$, and $\eta_m = 20$ used in [11]. Results after a single run are shown in Figures 3–5. Tests were repeated 100 times and the same metrics were measured as for the CTP problems earlier.[2] Obtained values are reported in Table 3. GDE3 outperforms NSGA-II with these problems according to metrics.

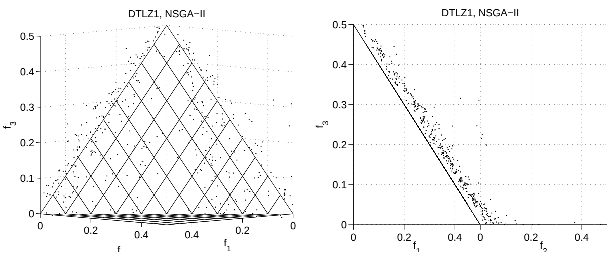

Figure 4: Two projections of the result for the DTLZ1 problem solved with NSGA-II.

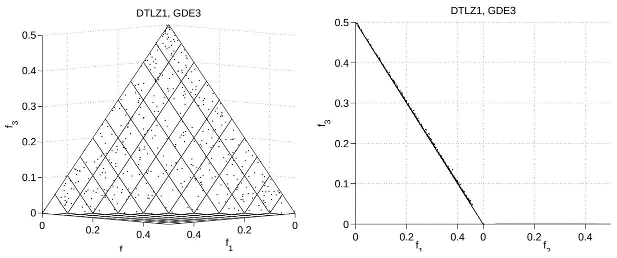

Figure 3: Two projections of the result for the DTLZ1 problem solved with GDE3.

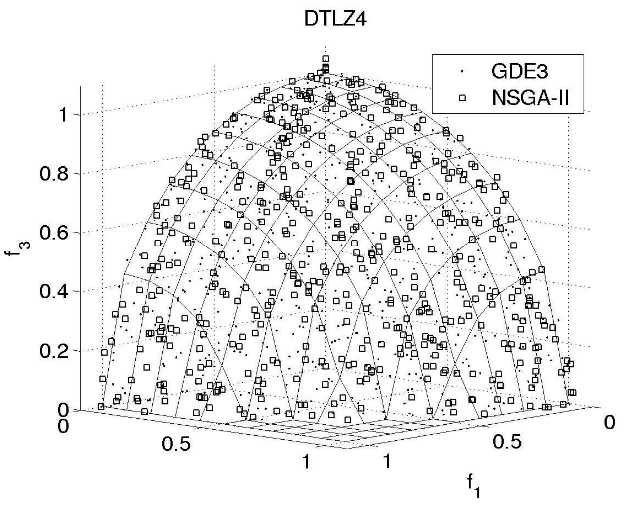

Figure 5: The DTLZ4 problem solved with GDE3 and NSGA-II.

6 Conclusions and Future Research

The third evolution version of Generalized Differential Evolution, GDE3, is proposed. GDE3 is designed for any number of objectives and constraints without introducing any extra control parameters to the original DE. In the case

[2]Even thought spacing might not give reliable result when the number of objectives is greater that two [11].

of unconstrained single-objective optimization problems, GDE3 is exactly the same as the original DE.

GDE3 modifies earlier GDE versions using a growing population and non-dominated sorting with pruning of non-dominated solutions to decrease the population size at the end of each generation. This improves obtained diversity and makes the method more stable for the selection of control parameter values. The constraint handling method used in GDEs reduces the number of needed function evaluations.

GDE3 was tested with a set of different types of test problems and results show an improved diversity of the solution over the NSGA-II method as well as demonstrating a reduction in the number of needed function evaluations. In some test problems, GDE3 found also a better converged solution. However, results are based on limited tests with a limited number of test problems and they are mainly indicative.

A more extensive comparison of GDE3 with other multi-objective DE methods, latest multi-objective evolutionary algorithms and test problems, parallelization of the algorithm, and applying GDE3 for practical constrained multi-objective problems remains as future work.

Acknowledgments

S. Kukkonen gratefully acknowledges support from the East Finland Graduate School in Computer Science and Engineering (ECSE), the Academy of Finland, Technological Foundation of Finland, and Finnish Cultural Foundation. He also wishes to thank Dr. K. Deb and all the KanGAL people for help and fruitful discussions.

Bibliography

[1] H. A. Abbass. The self-adaptive Pareto Differential Evolution algorithm. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002)*, pages 831–836, Honolulu, Hawaii, May 2002.

[2] H. A. Abbass, R. Sarker, and C. Newton. PDE: a Pareto-frontier Differential Evolution approach for multi-objective optimization problems. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC 2001)*, pages 971–978, Seoul, South Korea, 2001.

[3] B. V. Babu and M. M. L. Jehan. Differential Evolution for multi-objective optimization. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*, pages 2696–2703, Canberra, Australia, Dec 2003.

[4] P. K. Bergey. An agent enhanced intelligent spreadsheet solver for multi-criteria decision making. In *Proceedings of the Fifth Americas Conference on Information Systems (AMCIS 1999)*, pages 966–968, Milwaukee, Wisconsin, USA, Aug 1999.

[5] C. S. Chang and D. Y. Xu. Differential Evolution based tuning of fuzzy automatic train operation for mass rapid transit system. *IEE Proceedings on Electric Power Applications*, 147(3):206–212, May 2000.

[6] C. S. Chang, D. Y. Xu, and H. B. Quek. Pareto-optimal set based multiobjective tuning of fuzzy automatic train operation for mass transit system. *IEE Proceedings on Electric Power Applications*, 146(5):577–583, Sept 1999.

[7] T.-T. Chang and H.-C. Chang. Application of Differential Evolution to passive shunt harmonic filter planning. In *8th International Conference On Harmonics and Quality of Power*, pages 149–153, Athens, Greece, Oct 1998.

[8] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, England, 2001.

[9] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[10] K. Deb, A. Pratap, and T. Meyarivan. Constrained test problems for multi-objective evolutionary optimization. In *Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001)*, pages 284–298, Zurich, Switzerland, March 2001.

[11] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002)*, pages 825–830, Honolulu, Hawaii, May 2002.

[12] K. Deb and S. Tiwari. Omni-optimizer: A procedure for single and multi-objective optimization. In *Proceedings of the 3rd International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, pages 47–61, Guanajuato, Mexico, March 2005.

[13] R. J. G. Feng Xue, Arthur C. Sanderson. Pareto-based multi-objective Differential Evolution. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*, pages 862–869, Canberra, Australia, Dec 2003.

[14] V. Feoktistov and S. Janaqi. New strategies in Differential Evolution. In *Proceedings of the 6th International Conference on Adaptive Computing in Design and Manufacture (ACDM2004)*, pages 335–346, Bristol, United Kingdom, April 2004.

[15] J. E. Fieldsend, R. M. Everson, and S. Singh. Using unconstrained elite archives for multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 7(3):305–323, June 2003.

[16] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation sistributions in Evolution Strategies: The covariance matrix adaptation. In *Proceedings of the 1996 Congress on Evolutionary Computation (CEC 1996)*, pages 312–317, Nayoya University, Japan, May 1996.

[17] A. Inorio and X. Li. Solving rotated multi-objective optimization problems using Differential Evolution. In *Proceedings of the 17th Australian Joint Conference on Artificial Intelligence (AI 2004)*, pages 861–872, Cairns, Australia, Dec 2004.

[18] M. T. Jensen. Reducing the run-time complexity of multiobjective EAs: The NSGA-II and other algorithms. *IEEE Transactions on Evolutionary Computation*, 7(5):503–515, Oct 2003.

[19] S. Kukkonen and K. Deb. Improved pruning of non-dominated solutions based on crowding distance. Unpublished manuscript, 2005.

[20] S. Kukkonen and J. Lampinen. Comparison of Generalized Differential Evolution algorithm to other multi-objective evolutionary algorithms. In *Proceedings of the 4th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS2004)*, page 445, Jyväskylä, Finland, July 2004.

[21] S. Kukkonen and J. Lampinen. An extension of Generalized Differential Evolution for multi-objective optimization with constraints. In *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, pages 752–761, Birmingham, Finland, Sept 2004.

[22] J. Lampinen. DE's selection rule for multiobjective optimization. Technical report, Lappeenranta University of Technology, Department of Information Technology, 2001. [Online] Available: http://www.it.lut.fi/kurssit/03-04/010778000/MODE.pdf, 1.4.2005.

[23] J. Lampinen. A constraint handling approach for the Differential Evolution algorithm. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002)*, pages 1468–1473, Honolulu, Hawaii, May 2002.

[24] M. Laumanns, E. Zitzler, and L. Thiele. A unified model for multi-objective evolutionary algorithms with elitism. In *Proceedings of the 2000 Congress on Evolutionary Computation (CEC 2000)*, volume 1, pages 46–53, Piscataway, NJ, 2000.

[25] Y.-C. Lin, K.-S. Hwang, and F.-S. Wang. Hybrid Differential Evolution with multiplier updating method for nonlinear constrained optimization problems. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002)*, pages 872–877, Honolulu, Hawaii, May 2002.

[26] N. K. Madavan. Multiobjective optimization using a Pareto Differential Evolution approach. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002)*, pages 1145–1150, Honolulu, Hawaii, May 2002.

[27] E. Mezura-Montes, C. A. Coello Coello, and E. I. Tun-Morales. Simple feasibility rules and Differential Evolution for constrained optimization. In *Proceedings of the 3rd Mexican International Conference on Artificial Intelligence (MICAI 2004)*, pages 707–716, Mexico City, Mexico, April 2004.

[28] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, 1998.

[29] K. E. Parsopoulos, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis. Vector evaluated Differential Evolution for multiobjective optimization. In *Proceedings of the 2004 Congress on Evolutionary Computation (CEC 2004)*, pages 204–211, Portland, Oregon, USA, June 2004.

[30] K. V. Price. *New Ideas in Optimization*, chapter An Introduction to Differential Evolution, pages 79–108. McGraw-Hill, London, 1999.

[31] K. V. Price, R. Storn, and J. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization*. Springer-Verlag, Berlin, 2005.

[32] T. Robič and B. Filipič. DEMO: Differential Evolution for multiobjective optimization. In *Proceedings of the 3rd International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, pages 520–533, Guanajuato, Mexico, March 2005.

[33] R. Storn. System design by constraint adaptation and Differential Evolution. *IEEE Transactions on Evolutionary Computation*, 3(1):22–34, April 1999.

[34] R. Storn and K. V. Price. Differential Evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, Dec 1997.

[35] F.-S. Wang and J.-P. Chiou. Differential Evolution for dynamic optimization of differential-algebraic systems. In *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC'97)*, pages 531–536, Indianapolis, IN, 1997.

[36] F.-S. Wang and J.-W. Sheu. Multiobjective parameter estimation problems of fermentation processes using a high ethanol tolerance yeast. *Chemical Engineering Science*, 55(18):3685–3695, Sept 2000.

[37] D. Zaharie. Multi-objective optimization with adaptive Pareto Differential Evolution. In *Proceedings of Symposium on Intelligent Systems and Applications (SIA 2003)*, Iasi, Romania, Sept 2003.