Constrained Optimization based on a Multiobjective Evolutionary Algorithm

Anders Angantyr
Dept. of Applied Physics and Mechanical Eng.
Luleå University of Technology
SE-97187 Luleå
anders.angantyr@cad.luth.se

Johan Andersson
Dept. of Mechanical Engineering
Linköping University
SE-581 83 Linköping
johan@ikp.liu.se

Jan-Olov Aidanpaa
Dept. of Applied Physics and Mechanical Eng.
Luleå University of Technology
SE-97187 Luleå
joa@cad.luth.se

Abstract - A criticism of Evolutionary Algorithms (EAs) might be the lack of efficient and robust generic methods to handle constraints. The most widespread approach for constrained search problems is to use penalty methods. EAs have received increased interest during the last decade due to the ease of handling multiple objectives. A constrained optimization problem or an unconstrained multiobjective problem may in principle be two different ways to pose the same underlying problem. In this paper an alternative approach for the constrained optimization problem is presented. The method is a variant of a multiobjective real coded Genetic Algorithm (GA) inspired by the penalty approach. It is evaluated on six different constrained single objective problems found in the literature. The results show that the proposed method performs well in terms of efficiency, and that it is robust for a majority of the test problems.

1 Introduction

During the last decades Evolutionary Algorithms (EAs) have proved to become an important tool for difficult search and optimization problems. Most real-world problems are however constrained and a possible criticism of EAs has been the lack of efficient and generic constraint handling techniques. A comprehensive survey of existing constraint handling methods for EAs is done by Coello Coello in [1]. The frequently most used methods are based on various penalty functions for which some guidelines are given in [2]. Penalty methods are generic but may however distort the cost surface and introduce false optima. Most penalty methods also require additional parameters, which are problem-dependent and increase the complexity of the problem.

The constrained optimization problem may be handled as a multiobjective optimization problem as indicated by Coello Coello in [3], Michalewicz in [4] and Fonseca and Fleming in [5]. Furthermore, EAs based on non-dominated sorting for multiobjective problems have received increased interest during the past decade. Therefore it seems natural to look upon the constrained optimization problem as a multiobjective problem. Multiobjective approaches of constrained problems based on Shaffers VEGA [6] is found in [7] and [8]. Another interesting constraint handling method based on non-domination is presented by Deb et al. in [9]. To directly apply a multiobjective EA based on non-domination on a constrained optimization problem leads to a search of the best compromises of the objective value and constraint satisfaction. This whole set of solutions is usually not interesting since it is the optimal and feasible solution that is searched. Therefore it will not be efficient to directly apply a multiobjective EA on a constrained problem. Still the idea to handle the constrained problem with some variant of a multiobjective EA is interesting.

One of the most crucial steps in a multiobjective EA is how to rank individuals. In this paper an alternative ranking scheme for the constrained single objective problem is introduced. This ranking scheme is generic and no new parameters are introduced. The ideas of the ranking scheme are borrowed from the non-domination ranking for multiple objectives by Goldberg in [10] and penalty based methods for constrained problems.

The paper first defines the constrained optimization problem, and thereafter the proposed method is presented in more detail. Then the performance for a real coded Genetic Algorithm with the proposed ranking scheme implemented is tested on six different test problems used by Michalewicz in [11] and Deb in [12]. Finally, the result for this proposed method is compared to the result for other methods evaluated in [11] and [12].

2 The constrained optimization problem

In this section the constrained optimization problem and its terminology is defined. The constrained optimization problem or non-linear programming problem (NLP) with k inequality constraints and m equality constraints is formulated as

$$\begin{aligned} &\textit{Minimize} \quad f(\mathbf{x}) \\ &\textit{subject to} \\ &g_i(\mathbf{x}) \le 0 \quad i = 1,\ldots,k, \\ &h_i(\mathbf{x}) = 0 \quad i = 1,\ldots,m. \end{aligned} \tag{1}$$

$\mathbf{x} = [x_1, x_2, \ldots, x_n]$ is a vector of the n design variables such that $\mathbf{x} \in S \subseteq \Re^n$. The search space S is here defined as an n-dimensional rectangle by the upper and lower bounds for the design variables, $x_i^l \le x_i \le x_i^u \quad i = 1\ldots n$. The feasible region $F \subseteq S$ is the region of S for which the inequality and equality constraints are satisfied. The optimal solution is denoted $\mathbf{x}^*$. A constraint is said to be active at the point $\mathbf{x}^*$ if $g_i(\mathbf{x}^*) = 0$. By default all equality constraints are active at all points of the feasible space. Equality constraints may be transformed to inequality constraints [1] via

$$|h_i(\mathbf{x})| - \varepsilon \le 0 \tag{2}$$

where ε is a small tolerance. Since the algorithm that will be discussed does not use gradient information it does not matter if (2) is non-differentiable.

3 The proposed EA approach for constrained optimization

In this section the proposed ranking scheme is introduced. The non-dominated ranking by Goldberg [10] is used in a new way to formulate a scalar valued function that is used to rank individuals in the current population. Then selection, crossover, mutation and reinsertion are used in a standard manner for a real coded GA in this paper. This is described later since the focus for this section is to define the ranking scheme.

It is first assumed that all equality constraints are transformed by (2) so the problem is now

$$\begin{aligned} &\textit{Minimize} \quad f(\mathbf{x}) \\ &\textit{subject to} \\ &g_i(\mathbf{x}) \le 0 \quad i = 1,\ldots, p \end{aligned} \tag{3}$$

where $p = k + m$.

Now, the objective function is given index 1, $f_1(\mathbf{x}) = f(\mathbf{x})$. Then the constraints $g_i(\mathbf{x})$ are reformulated into new objectives $f_{1+i}(\mathbf{x})$. These objectives are defined as

$$f_{1+i}(\mathbf{x}) = \max\left(0, g_i(\mathbf{x})\right), \quad i = 1,\ldots, p \tag{4}$$

A natural approach would be to apply a Pareto based multiobjective GA to solve the problem. This might not be the best idea since the Pareto optimal set with respect to the new objectives $f_1(\mathbf{x})$ to $f_{p+1}(\mathbf{x})$ is generally not the same as the optimal solution $\mathbf{x}^*$. The idea here is to treat the objective $f_1(\mathbf{x})$ and the objectives $f_2(\mathbf{x})$ to $f_{p+1}(\mathbf{x})$ separately. The approach is based on the following criteria

- If no feasible individual exists in the current population, the search should be directed towards the feasible region.
- If a majority of the individuals in the current population are feasible, the search should be directed towards the unconstrained optimum.
- A feasible individual closer to the optimum is always better than a feasible individual further from the optimum.
- An infeasible individual might be a better individual than a feasible individual if the number of feasible individuals is high.

From the above statements it is clear that the search direction should be dependent upon the number of feasible individuals in the current population. The reason for the fourth statement is that an infeasible individual with a good objective value ($f_1(\mathbf{x})$) should not be rejected as it might guide the search towards the true optimum by improving the diversity of the population.

Now P is defined to be the population size and N the number of feasible solutions in the current population. $\mathbf{x}_j$ is the j^{th} individual in the current population. Then, $rank_1(\mathbf{x}_j)$ is defined as the ranking according to the first objective $f_1(\mathbf{x})$. The best individual gets $rank_1 = 1$.

$rank_2(\mathbf{x}_j)$ is defined to be the non-dominated ranking with respect to $f_2(\mathbf{x})$ to $f_{p+1}(\mathbf{x})$ as defined by Goldberg [10]. In the ranking the first non-dominated individuals in the population receive $rank_2 = 1$. Then these individuals are removed from the population and the ranking is repeated for the remaining individuals, but now the non-dominated individuals get $rank_2 = 2$. This is repeated until all individuals in the current population have received a value for $rank_2$. In [9] Deb shows a method with computational complexity $O((p+1)\mathrm{P}^2)$ to perform the non-dominated ranking.

Now a new objective function $\phi(\mathbf{x}_j)$ is formulated as

$$\phi(\mathbf{x}_j) = \frac{\mathrm{N}}{\mathrm{P}} rank_1(\mathbf{x}_j) + \frac{\mathrm{P} - \mathrm{N}}{\mathrm{P}} rank_2(\mathbf{x}_j). \tag{5}$$

Each individual is then ranked according to its value for Equation 5 and fitness is assigned in a regular manner. Note that if no feasible solution is present in the population (N = 0), the ranking according to the objective ($rank_1$) becomes inactive and the population is ranked according to the constraints ($rank_2$), i.e. the search is guided towards the feasible region. On the other hand, if all individuals are feasible (N = P), the population is ranked according to the objective ($rank_1$), and the search is directed towards the unconstrained optimum. Among two feasible individuals, the most fit is the one with lower value for $rank_1$ (the objective) since all feasible individuals receive $rank_2 = 1$. All these observations are consistent with the previously listed criteria.

An interesting feature for the new ranking is that the search direction depends on the number of feasible solutions. If many feasible solutions exist, the search is directed towards the unconstrained optimal solution. If now it is assumed that the unconstrained optimum is located outside the feasible region, the population may tend to oscillate over the boundary to the feasible region. This variation of the search direction gives a positive effect of the diversity in the population.

Equation 5 has a similar structure as a penalty based approach but it should be pointed out that no parameter that requires problem dependent fine-tuning is introduced. The "weights" for the two objectives in Equation 5 only depend on the population size and the number of feasible individuals in the current population.

The new ranking procedure for a NLP problem is summarized below

1. Reformulate the problem according to Equation 3 and Equation 4
2. Rank the population with respect to the objective ($f_1(\mathbf{x})$) and assign it to $rank_1$
3. Rank the population with respect to the constraints ($f_2(\mathbf{x})$ to $f_{p+1}(\mathbf{x})$) based on non-dominance according to Goldberg [10] and assign it to $rank_2$
4. Calculate the objective ($\phi(\mathbf{x})$) according to Equation 5

5. Rank the population according to the single objective $\phi(\mathbf{x})$

Until now, only the ranking has been described. This ranking scheme may be used with any type of GA. In the rest of this paper a real coded GA with the proposed ranking scheme is used. All the GA operations and parameters are chosen as simple as possible. Therefore a more advanced algorithm, such as an adaptive GA for example, might improve the results presented in this paper. Linear fitness assignment according to the ranking for the new objective (Equation 5) is used. The selective pressure is set to 1.9. The selection method is the roulette wheel selection. The number of selected individuals are defined by the generation gap that is set to 95%. Thus 95% of the population is selected for mating and the worst parents are replaced by all the offspring. Hence, an elitist GA is used. Blend crossover, BLX, see [13], is used with a probability equal to 1. The mutation operator by Mühlenbein and Schlierkamp-Voosen [14] which produces a small mutation step with high probability and a large step with small probability is used. The mutation probability is set to $1/n$ where n is the number of design variables. The maximum mutation step is defined in the result section.

4 An illustrative example

In this section the ranking based on Equation 5 is discussed for a simple NLP problem. It should be clear that the purpose of this section is to show the important effects of the ranking and not to solve the simple NLP problem. First the result of an actual search is presented. Then the imposed search direction is discussed with the help of two hypothetical populations.

The problem is as follows. A quadratic function is to be minimized and the feasible solutions are constrained by three circles. The problem is stated as

$$
\begin{aligned}
&\textit{Minimize} \quad f(\mathbf{x}) = x_1^{\,2} + x_2^2 \\
&\textit{subject to} \\
&g_1(\mathbf{x}) \equiv x_1^2 + (x_2 - 3)^2 \le 1.5^2, \\
&g_2(\mathbf{x}) \equiv (x_1 - 1)^2 + (x_2 - 1)^2 \le 1.5^2, \\
&g_3(\mathbf{x}) \equiv (x_1 + 1)^2 + (x_2 - 1)^2 \le 1.5^2, \\
&-5 \le x_1 \le 5, \\
&-3 \le x_2 \le 7.
\end{aligned}
\tag{6}
$$

For the unconstrained problem the optimal solution is $\mathbf{x}^* = [0, 0]$. For the constrained problem (6) the optimal solution is $\mathbf{x}^* = [0, 1.5]$. The first constraint is active at the optimal solution. The population size is set to 10, the maximum number of generations is 50 and the maximum mutation step is set to 0.1 of the range for the design variables. The mutation probability is set to 0.2 in this case.

The initial and the final generation are shown in Figure 1. The rank of the initial generation according to Equation 5 is also given in the figure.

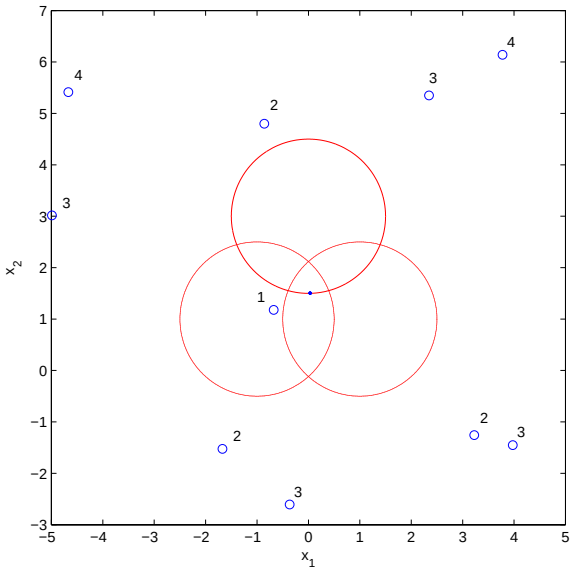

Figure 1: Initial generation (the smaller rings) and final generation (the "dot" near the optimum) shown in design variable space. The numbers indicate the rank according to Equation (5).

The best individual in the initial generation correspond to ranking 1. Figure 1 shows clearly that the search direction is towards the feasible region in the initial generation.

Figure 2 shows the ratio of feasible solutions, the mean normalized Euclidian distance and the ratio between the true optimum and the best-found feasible objective value. To avoid premature convergence it is crucial to have sufficient diversity in the population. An indication of the diversity in the population is given by the distance between the members of the population. The distance between two individuals is calculated using the normalized Euclidian distance. The mean Euclidian distance is obtained by calculating the mean distance between all individuals in the population, and hence is a measure of the diversity in the population.

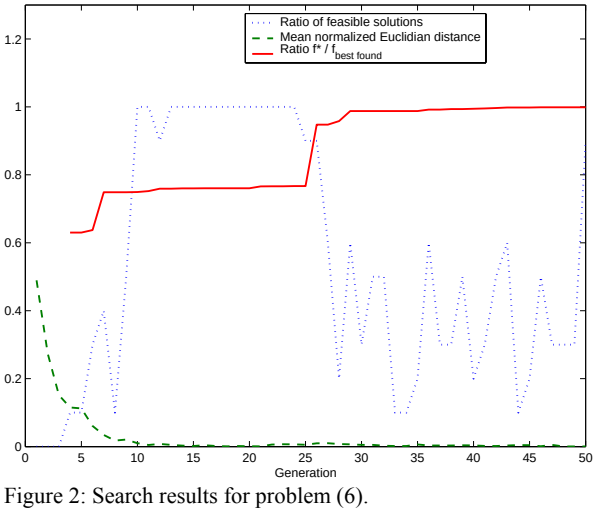

Figure 2: Search results for problem (6).

The first feasible solution is found in generation 4. In the early generations (~ 5 to 10) the number of feasible individuals increases rapidly. In generations 13 to 24 all individuals are feasible and the improvement in the

objective function is very small since the population has converged too fast. In generation 25 an infeasible individual is created by a mutation. This individual is better than all the feasible individuals in terms of the objective value, $f(\mathbf{x})$. Due to the "weights" in Equation 5 this infeasible individual becomes the best individual. The search is then directed out of the feasible region towards the unconstrained optimum and as a result better feasible solutions are found. To make this variation of the search direction more clear two populations with different ratio of feasible individuals are studied in Figure 3 and Figure 4.

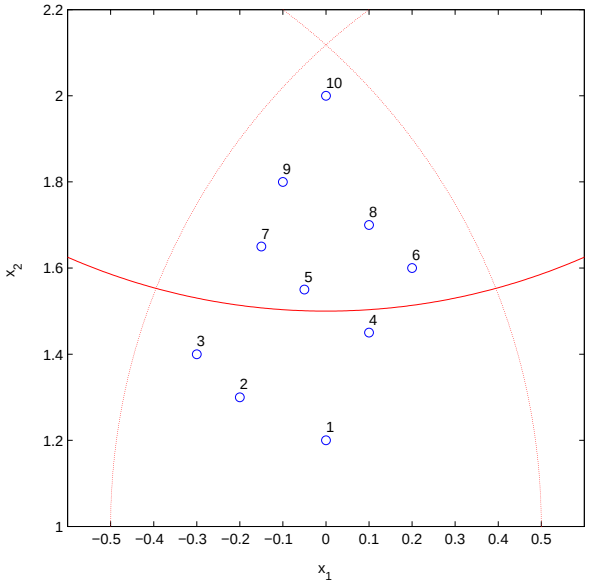

Figure 3. Rank according to Equation 5 for a population with 60% feasible individuals.

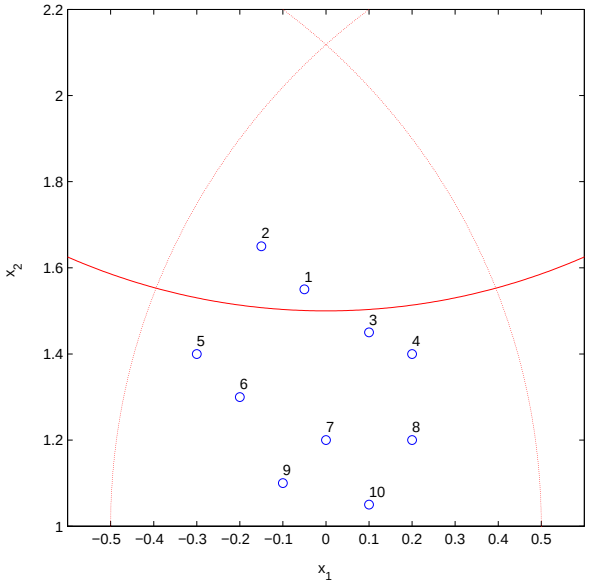

Figure 4. Rank according to Equation 5 for a population with 20% feasible individuals.

When the number of feasible individuals is high, the search is directed towards the unconstrained optimum as shown in Figure 3. In the later stage, when more infeasible individuals occur in the population the search is directed back to the feasible region again. This explains the oscillating behaviour of the ratio of feasible individuals for generation 26 to 50 in Figure 1.

This example shows the dynamic behaviour of the search direction. The oscillation of the search direction only occurs when at least one constraint is active at the optimum. The variation of the search direction has a positive effect for the population diversity. Thus, if mutation is used no special operation to preserve the diversity in the population is required for most cases. In the next section the method is evaluated using a set of selected test problems gathered from the literature.

5 Constrained single objective test problems

In [15], Michalewicz et al. present a test case generator to use in tests of algorithms for constrained optimization problems. This test case generator will probably be used in future research on constrained optimization problems. The results for different constraint handling methods are yet quite limited for this test case generator. Therefore a set of test problems for which there exists results for many different algorithms is here chosen instead.

It is always difficult to make fair comparisons between different EAs. Two different strategies may well have different optimal settings for the optimization algorithm parameters on the same problem. Another difficulty is to determine how to compare different algorithms. A naive but obvious way to compare algorithms is to compare the best solution found in the same number of function evaluations. A measure of the robustness of the algorithms is indicated by the spread of the best solutions found if the optimization is run several times independently. Here it is chosen to compare the results for the proposed ranking scheme with previously reported results for other EAs by other authors on a set of problems. Six test problems are selected. Problem #2 to #6 are found in [11] and problem #1 to #6 in [12]. A short summary of the test problems is given in Table 1. The size of the feasible region is estimated by the ratio (ρ) of feasible solutions found in a random sampling of 10^6 solutions in the search space[1]. The six test problems are described in detail in the next subsections.

Table 1: Summary of test problems. C corresponds to the number of constraints, A to the number of active constraints at the optimum and n is the number of design variables.

Problem	n	Type of f	ρ	C	A
#1	5	quadratic	52.03%	6	2
#2	13	quadratic	0.0111%	9	6
#3	8	linear	0.0010%	6	6
#4	7	polynomial	0.5121%	4	2
#5	5	nonlinear	0.0000%	3	3
#6	10	quadratic	0.0003%	8	6

In [11], Michalewicz compares the performance of six different methods on the five problems #2 to #6. Most of the methods are based on penalty functions, The result here is compared to the result for the best method found in [11]. In [12], Deb proposes a special penalty based method

[1] The ratio for problem #2 to problem #6 is presented in [11].

for which the following criteria are always enforced if a tournament selection operator is used:

1. Any feasible solution is preferred to any infeasible solution.
2. Among two feasible solutions, the one having better objective function value is preferred.
3. Among two infeasible solutions, the one having smaller constraint violation is preferred.

Deb tested his method on nine different problems of which test problem #1 to test problem #6 are a subset. The results obtained by the proposed method in this paper are compared to the results obtained by Deb on all these six test problems. Furthermore, Deb stated that *"In all cases, the proposed approach has been able to repeatedly find solutions closer to the true optimum solution than that reported earlier"*. Therefore a fair comparison to the results reported in [12] should give good indication of the performance of the method presented in this paper.

It is worth to notice that the effect of the ranking scheme introduced in this paper is similar to the above listed criteria only if there exist few feasible individuals in the current population. On the contrary, if there exist many feasible individuals, a good (in terms of the objective value) infeasible individual may well be preferred to a feasible individual if this is worse in terms of the objective value.

5.1 Test problem 1

This problem was first presented by Himmelblau in [16]. It has later been used by Coello Coello [1] and Deb [12] to evaluate the performance of various GAs for constrained optimization. The problem is stated as

$$
\begin{aligned}
&\textit{Minimize} \quad f(\mathbf{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141\\
&\textit{subject to}\\
&g_1(\mathbf{x}) \equiv 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 \ge 0,\\
&g_2(\mathbf{x}) \equiv 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 \le 92,\\
&g_3(\mathbf{x}) \equiv 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 \ge 90,\\
&g_4(\mathbf{x}) \equiv 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 \le 110,\\
&g_5(\mathbf{x}) \equiv 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 \ge 20,\\
&g_6(\mathbf{x}) \equiv 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 \le 25,\\
&78 \le x_1 \le 102,\\
&33 \le x_2 \le 45,\\
&27 \le x_i \le 45, \quad i = 3,4,5.
\end{aligned}
$$

The best-known solution to this problem [16] is $\mathbf{x}^*$ = [78, 33, 29.995, 45, 36.776] which gives f^* = -30665.5. At this solution the constraints g_2 and g_5 are active [12].

5.2 Test problem 2

The problem is stated as follows

$$
\begin{aligned}
&\textit{Minimize} \quad f(\mathbf{x}) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i\\
&\textit{subject to}\\
&g_1(\mathbf{x}) \equiv 2x_1 + 2x_2 + x_{10} + x_{11} \le 10,\\
&g_2(\mathbf{x}) \equiv 2x_1 + 2x_3 + x_{10} + x_{12} \le 10,\\
&g_3(\mathbf{x}) \equiv 2x_2 + 2x_3 + x_{11} + x_{12} \le 10,\\
&g_4(\mathbf{x}) \equiv -8x_1 + x_{10} \le 0,\\
&g_5(\mathbf{x}) \equiv -8x_2 + x_{11} \le 0,\\
&g_6(\mathbf{x}) \equiv -8x_3 + x_{12} \le 0,\\
&g_7(\mathbf{x}) \equiv -2x_4 - x_5 + x_{10} \le 0,\\
&g_8(\mathbf{x}) \equiv -2x_6 - x_7 + x_{11} \le 0,\\
&g_9(\mathbf{x}) \equiv -2x_8 - x_9 + x_{12} \le 0,\\
&0 \le x_i \le 1, \quad i = 1,...,9,\\
&0 \le x_i \le 100, \quad i = 10,11,12,\\
&0 \le x_{13} \le 1.
\end{aligned}
$$

The optimal objective value for this problem is f^* = -15 for x* = [1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1]. At this solution all constraints except g_4, g_5 and g_6 are active.

5.3 Test problem 3

The third test problem is

$$
\begin{aligned}
&\textit{Minimize} \quad f(\mathbf{x}) = x_1 + x_2 + x_3\\
&\textit{subject to}\\
&g_1(\mathbf{x}) \equiv 1 - 0.0025(x_4 + x_6) \ge 0,\\
&g_2(\mathbf{x}) \equiv 1 - 0.0025(x_5 + x_7 - x_4) \ge 0,\\
&g_3(\mathbf{x}) \equiv 1 - 0.01(x_8 - x_5) \ge 0,\\
&g_4(\mathbf{x}) \equiv x_1x_6 - 833.33252x_4 - 100x_1 + 83333.333 \ge 0,\\
&g_5(\mathbf{x}) \equiv x_2x_7 - 1250x_5 - x_2x_4 + 1250x_4 \ge 0,\\
&g_6(\mathbf{x}) \equiv x_3x_8 - x_3x_5 + 2500x_5 - 1250000 \ge 0,\\
&100 \le x_1 \le 10000,\\
&1000 \le x_i \le 10000, \quad i = 2,3,\\
&10 \le x_i \le 1000, \quad i = 4,...,8.
\end{aligned}
$$

The optimum solution is **x*** = [579.3167, 1359.943, 5110.071, 182.0174, 295.5985, 217.9799, 286.4162, 395.5979] which gives f^* = 7049.330923. All six constraints are active at the optimal solution.

5.4 Test problem 4

This problem is stated as

$$
\begin{aligned}
&\textit{Minimize} \quad f(\mathbf{x}) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 +\\
&\qquad 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7\\
&\qquad -10x_6 - 8x_7\\
&\textit{subject to}\\
&g_1(\mathbf{x}) \equiv 127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \ge 0,\\
&g_2(\mathbf{x}) \equiv 282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \ge 0,\\
&g_3(\mathbf{x}) \equiv 196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \ge 0,\\
&g_4(\mathbf{x}) \equiv -4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \ge 0,\\
&-10 \le x_i \le 10, \quad i = 1,...,7.
\end{aligned}
$$

The optimal solution is $\mathbf{x}^*$ = [2.330499, 1.951372, -0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227] which gives f^* = 680.6300573. The constraints g_1 and g_4 are active at the optimal solution.

5.5 Test problem 5

The fifth test problem is stated as

$$\begin{aligned}
&\textit{Minimize} \quad f(\mathbf{x}) = e^{x_1 x_2 x_3 x_4 x_5} \\
&\textit{subject to} \\
&h_1(\mathbf{x}) \equiv \sum_{i=1}^{5} x_i^2 = 10, \\
&h_2(\mathbf{x}) \equiv x_2 x_3 - 5 x_4 x_5 = 0, \\
&h_3(\mathbf{x}) \equiv x_1^3 + x_2^3 = -1, \\
&-2.3 \le x_i \le 2.3, \quad i = 1,2, \\
&-3.2 \le x_i \le 3.2, \quad i = 3,4,5.
\end{aligned}$$

The optimal solution is $\mathbf{x}^*$ = [-1.717143, 1.595709, 1.827247, -0.7636413, -0.7636450]. This gives f^* = 0.053950. Since all constraints are equality type, all constraints are active at the optimal solution. The equality constraints are transformed into inequality constraints by Equation 2 and the tolerance is set to ε = 0.001 for the results presented in this paper.

5.6 Test problem 6

The last test problem is

$$\begin{aligned}
&\textit{Minimize} \quad f(\mathbf{x}) = x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 \\
&\qquad + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45 \\
&\textit{subject to} \\
&g_1(\mathbf{x}) \equiv 105 - 4x_1 - 5x_2 + 3x_7 - 9x_8 \ge 0, \\
&g_2(\mathbf{x}) \equiv -10x_1 + 8x_2 + 17x_7 - 2x_8 \ge 0, \\
&g_3(\mathbf{x}) \equiv 8x_1 - 2x_2 - 5x_9 + 2x_{10} + 12 \ge 0, \\
&g_4(\mathbf{x}) \equiv -3(x_1 - 2)^2 - 4(x_2 - 3)^2 - 2x_3^2 + 7x_4 + 120 \ge 0, \\
&g_5(\mathbf{x}) \equiv -5x_1^2 - 8x_2 - (x_3 - 6)^2 + 2x_4 + 40 \ge 0, \\
&g_6(\mathbf{x}) \equiv -x_1^2 - 2(x_2 - 2)^2 + 2x_1 x_2 - 14x_5 + 6x_6 \ge 0, \\
&g_7(\mathbf{x}) \equiv -0.5(x_1 - 8)^2 - 2(x_2 - 4)^2 - 3x_5^2 + x_6 + 30 \ge 0, \\
&g_8(\mathbf{x}) \equiv 3x_1 - 6x_2 - 12(x_9 - 8)^2 + 7x_{10} \ge 0, \\
&-10 \le x_i \le 10, \quad i = 1,\ldots,10.
\end{aligned}$$

The optimal solution is x* = [2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927] which gives f^* = 24.3062091. All constraints except g_7 and g_8 are active at the optimal solution.

6 Results

First some typical search results for the first three problems are presented in Figure 5 to Figure 7. These figures show the ratio of feasible solutions, the mean normalized Euclidian distance and the ratio between the optimal solution and the best-found feasible solution. The GA parameters used in this study is presented in Table 3 for each problem. Then the result for this algorithm is compared to the best result reported in [11] and the result reported in [12].

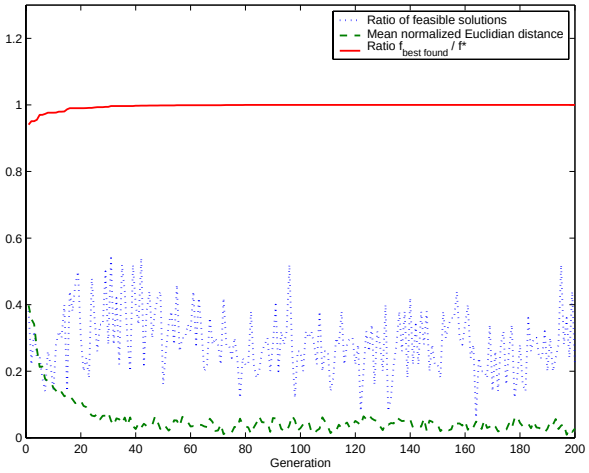

Figure 5: Typical result for problem #1.

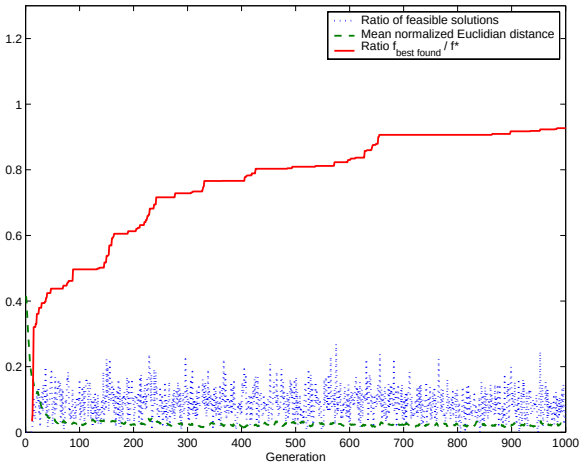

Figure 6: Typical result for problem #2.

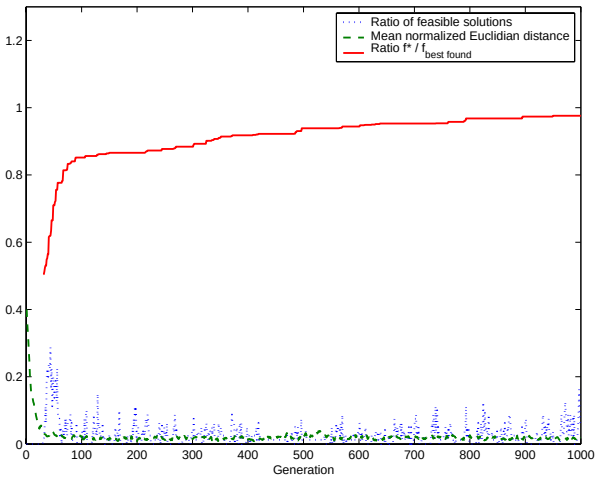

Figure 7: Typical result for problem #3.

Clearly the easiest problem is the first test problem. Near optimal solutions are found in early generations. Surprisingly the most difficult problem of these three problems for this method is test problem #2. In [11] it was reported that this was one of the easiest problem and only a few of the methods studied had any difficulties on this problem.

The results for this algorithm are now compared to the best results for all tested methods in [11] and summarized in Table 2. The population size is 70 and the maximum number of generations is 5000, both in this study and for all algorithms tested in [11]. For this algorithm the maximum mutation step is set to 0.1 of the range for the design variables. The result from [11] presented in Table 2 are the results for the method that found the best feasible solution. It should be mentioned that all the results in the coming tables correspond to feasible solutions.

Table 2: Result for this algorithm compared to best result in [11]. The number of independent runs is 10.

Problem	Study	Best	Median	Worst
#2	This study	-14.680	-14.570	-12.419
	Best in [11]	-15.000	-15.000	-15.000
#3	This study	7079.5	7107.0	7187.8
	Best in [11]	7378.0	8206.2	9653.0
#4	This study	680.636	680.640	680.646
	Best in [11]	680.642	680.718	680.955
#5	This study	0.313	0.534	0.602
	Best in [11]	0.054	0.064	0.577
#6	This study	24.519	24.600	24.735
	Best in [11]	25.486	26.905	42.358

As can be seen from Table 2 this algorithm finds better results in problem #3, problem #4 and problem #6 than all methods tested in [11]. The variation is also much less on these problems.

In Table 4 the result for this algorithm is compared to the results by Deb in [12]. Table 3 shows the GA parameters used for the results in Table 4.

Table 3: GA parameters used for the results presented in Table 4.

Problem	Study	Pop size	Max gen	Max mutation step
#1	This study	50	1000	0.1
	[12]	50	5000	-
#2	This study	130	2000	0.1
	[12]	130	N/A	-
#3	This study	80	4000	0.1
	[12]	80	4000	-
#4	This study	70	5000	0.02
	[12]	70	5000	-
#5	This study	50	7000	0.1
	[12]	50	7000	-
#6	This study	100	3500	0.02
	[12]	100	3500	-

Table 4: Result for this algorithm compared to best result in [12]. The independent number of runs is 50.

Problem	Study	Best	Median	Worst
#1	This study	-30665.5	-30665.5	-30665.4
	[12]	-30665.5	-30665.5	-29846.7
#2	This study	-14.276	-13.224	-11.963
	[12]	-15.000	-15.000	-13.000
#3	This study	7072.4	7100.2	7256.4
	[12]	7060.2	7220.0	10230.8
#4	This study	680.632	680.636	680.645
	[12]	680.634	680.642	680.651
#5	This study	0.44678	0.56967	0.83732
	[12]	0.05395	0.24129	0.50776
#6	This study	24.375	24.426	24.512
	[12]	24.372	24.409	25.075

The best found result of the 50 independent runs for this method is almost similar to the result reported by Deb for problem #1, problem #3, problem #4 and problem #6. For these problems the variation in the best results found is less for the proposed method than that reported in [12]. For problem #2 and problem #5 the method presented by Deb performs better, both in terms of best-found solution and variation of the best-found solution.

It should be mentioned however, that the results presented by Deb are based on tournament selection with a niching method that required two extra parameters. Furthermore, the maximum number of generations for the results of test problem #2 in [12] is not known. Hence it is difficult to make a fair comparison of the results on this problem.

7 Conclusions

A general ranking scheme without problem specific extra parameters for constrained optimization problem has been presented. The performance for an algorithm with this ranking scheme has also been compared to the result of other algorithms on six problems previously used by other authors. The results encourage further research since the method performs better than many other algorithms for the tested constrained single objective problems. It is also shown that the robustness in terms of minimum spread in best found solutions, is better than one of the best methods on a majority of the six tested problems. It was only in the problem containing equality constraints (problem #5) that this method did not perform well. It could not match up to the results for the other algorithms on problem #2 either. The cause of this is an open question for further research. It should also be mentioned that no effort has been made to study the optimal parameter settings such as population size, generation gap, mutation probability, etc. The performance of this ranking scheme may well be better in a more advanced GA, for example an adaptive GA.

An obvious extension to the presented ranking scheme is to address constrained multiobjective problems as well. By redefining $rank_1$ as the Pareto ranking presented by Fonseca and Fleming in [17], the presented ranking scheme could handle problems with multiple objectives. This is an area of ongoing research and the preliminary results are encouraging.

Bibliography

[1] Coello Coello, C. A., 2002, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of state of the art", Computer methods in applied mechanics and engineering, **191**, pp. 1245-1287.

[2] Richardson, J. T., Palmer, M. R., Liepins, G. and Hilliard ,M., 1989, "Some Guidelines for Genetic Algorithms with Penalty Functions", In J. D. Schaffer (Ed.), *Proceedings of the 3rd International Conference on Genetic Algorithms*, Morgan Kaufmann, Reading, MA, pp. 191-197.

[3] Coello Coello, C. A., 1999, "A Survey of Constraint Handling Techniques used with Evolutionary Algorithms", Technical Report Lania-

RI-99-04, Laboratorio Nacional de Informática Avanzada, Xalapa.

[4] Michalewicz, Z., 1995, "A Survey of Constraint Handling Techniques in Evolutionary Computation Methods", In J. R. McDonnell, R. G. Reynolds and D. B. Fogel (Eds.), *Proceedings of the 4th Annual Conference on Evolutionary Programming*, MIT Press, Cambridge, MA, pp. 135-155,

[5] Fonseca, C. M. and Fleming, P. J, 1995, "Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms I: A Unified Formulation", Research Report 564, University of Sheffield, Sheffield.

[6] Schaffer, J. D., 1985, "Multiple objective optimization with vector evaluated genetic algorithms", In J. J. Grefensttete (Ed.) *Proceedings of the 1st Int. Conference on Genetic Algorithms,* Hillsdale, New Jersey: Lawrence Erlbaum Associates, pp. 93-100.

[7] Surry, P., Radcliffe, N., Boyd, I., 1995, "A multi-objective approach to constrained optimization of gas supply networks", In T. C. Fogarty (Ed.), *Proceedings of the AISB-95 Workshop on Evolutionary Computing*, Springer, Sheffield, pp. 166-180.

[8] Parmee, I. C. and Purchase, G., 1994, "The development of a directed genetic search technique for heavily constrained search spaces", In I. C. Parmee (Ed.), *Proceedings of the Conference on Adaptive Computing in Engineering Design and Control*, University of Plymouth, Plymouth, pp. 97-102.

[9] Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T., 2000, "A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II", KanGAL Report No. 200001, Kanpur Genetic Algorithm Laboratory, Kanpur.

[10] Goldberg, D. E., 1989, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading.

[11] Michalewicz, Z., 1995, "Genetic Algorithms, Numerical Optimization, and Constraints", In L. Eshelman (Ed.), *Proceedings of the 6th International Conference on Genetic Algorithms*, Morgan Kaufmann, San Francisco, pp. 151-158.

[12] Deb, K., 2000, "An efficient constraint handling method for genetic algorithms", Computer methods in applied mechanics and engineering, **186**, pp. 311-338.

[13] Eshelman L. J. and Schaffer J. D., 1993, "Real-Coded Genetic Algorithms and Interval-Schemata", In L. D. Whitley (Ed.), *Foundations of Genetic Algorithms 2*, Morgan Kaufmann, San Mateo, CA, pp. 187-202.

[14] Mühlenbein, H. and Schlierkamp-Voosen, D., 1993, "Predictive Models for the Breeder Genetic Algorithm: I. Continuous Parameter Optimization", Evolutionary Computation, **1**, pp. 25-49.

[15] Michalewicz, Z., Deb, K., Schmidt, M. and Stidsen, T. J., 1999, "Towards understanding constraint-handling methods in evolutionary algorithms", *In Proceedings of the 1999 Congress on Evolutionary Computation*, IEEE Service Centre, Washington DC, pp. 581-588.

[16] Himmelblau, D. M., 1972, *Applied Nonlinear Programming*, McGraw-Hill, New York.

[17] Fonseca, C. M. and Fleming, P. J., 1993, "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization", In S. Forrest (Ed.), *Proceedings of the 5th International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, pp. 416-423.